NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

Estimation of Motion Parameters from Image Sequences

by

Fatih Ildız

June, 1991

Thesis Advisor: Prof. Jeffrey B. Burl

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: Naval Postgraduate School; 6b. OFFICE SYMBOL (if applicable): EC	7a. NAME OF MONITORING ORGANIZATION: Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code): Monterey, CA 93943-5000	7b. ADDRESS (City, State, and ZIP Code): Monterey, CA 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION; 8b. OFFICE SYMBOL (if applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
ESTIMATION OF MOTION PARAMETERS FROM IMAGE SEQUENCES(U)

12. PERSONAL AUTHOR(S)
İLDİZ, Fatih

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Master's Thesis	FROM TO	1991, JUNE	95

16. SUPPLEMENTARY NOTATION: The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Extended Kalman Filter; Spatiotemporal Gradient; Spatiotemporal Frequency; One-dimensional FFT.

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The image motion analysis algorithms that generate the two-dimensional velocity of objects in a sequence of images are developed. The algorithms considered consist of: the extended Kalman filter method; the spatiotemporal gradient methods; the spatiotemporal frequency methods; and the one-dimensional FFT methods. These algorithms are designed to perform on low signal to noise ratio images. Each of these algorithms is applied to a sequence of computer generated images with varying signal to noise ratios. Simulations are used to evaluate performance of each algorithm.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: [X] UNCLASSIFIED/UNLIMITED [ ] SAME AS RPT. [ ] DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL: BURL, Jeffrey B.	22b. TELEPHONE (Include Area Code): (408) 646-2390; 22c. OFFICE SYMBOL: EC/Bl

Approved for public release; distribution is unlimited.

Estimation of Motion Parameters
from Image Sequences

by

Fatih Ildız
Lieutenant Junior Grade, Turkish Navy
B.S., Turkish Naval Academy, 1985

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

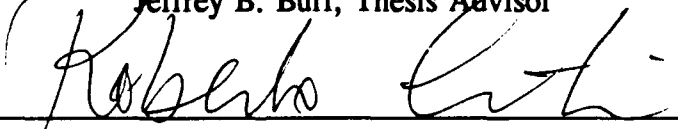
NAVAL POSTGRADUATE SCHOOL
June, 1991

Author: 

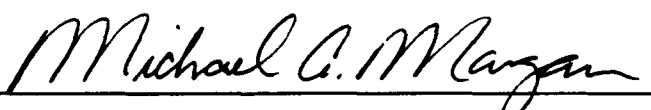
Fatih Ildız

Approved by: 

Jeffrey B. Burl, Thesis Advisor



Roberto Cristi, Second Reader



Michael A. Morgan, Chairman
Department of Electrical and Computer Engineering

ABSTRACT

The image motion analysis algorithms that generate the two dimensional velocity of objects in a sequence of images are developed. The algorithms considered consist of: the parallel extended Kalman filter method; the spatiotemporal gradient methods; the spatiotemporal frequency methods; and the one-dimensional FFT methods. These algorithms are designed to perform on low signal to noise ratio images. Each of these algorithms is applied to a sequence of computer generated images with varying signal to noise ratios. Simulations are used to evaluate the performance of each algorithm.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I would like to thank Professor J. B. Burl for his assistance, encouragement , and patience during the course of this study. I would also like to thank my classmates LTJG İ. Aksu, TK Navy, and LTJG E. Aykaç TK Navy, for their share and assistance. Finally, I would like to express my appreciation to my love D. Maras for her encouragement, support, and understanding during the many hours that this work required.

I. INTRODUCTION

The measurement of displacement between successive frames in an image sequence gives information about the motion (or velocity) of the objects in these images. The motion information is an important element in the source coding, restoration, analysis and interpretation of time-varying imagery, since a time-varying scene can be characterized to a large extent by the motion it contains. The characterization may lead to detection of objects (targets, their velocities and trajectories), it can be used to distinguish between scene and noise components in order to reduce additive noise in sequences and furthermore can be used as a cue for image segmentation. Motion imformation may also be useful for the compact representation of the image sequences for transmission and storage purposes (as in motion-compensated interframe coders). The motion patterns in the image sequence may be due to the motion of objects in the scene, the motion of the observer or both. Also, the motion may be apparent motion where a change in the image intensity between frames gives the illusion of motion.

There are 4 basic approaches for motion estimation:

(1) feature correspondence (matching),

(2) difference picture methods,

(3) spatio-temporal gradients,

(4) Fourier phase changes.

Matching methods use algorithms for matching the components of objects between two

successive frames of a scene containing moving objects. This leads to an optimization process which can easily be solved. The drawback to this approach is that identifiable features must be derived from the image before matching can be tried. Also the spatial range over which image flow is derived is short in comparison to the size of the region in which identifiable features exist. Difference picture methods start with a pixel-wise difference between successive frames in an image sequence. The differences can be thresholded to produce a binary motion image. Since difference picture methods do not take the direction or speed of the motion into account, the methods are not easily applied when the background is moving and can not easily differentiate between different objects moving with different velocities. The problem with difference picture methods is that they are local and do not combine spatial and temporal changes over neighborhoods to improve the information on the velocity field. Spatio-temporal gradient methods relate the spatial and temporal gradient at each pixel in the image plane to the instantaneous velocity at that pixel point. This forms a two-dimensional vector field called the displacement field (velocity field or optical flow). The Fourier-phase methods utilize the shift property of Fourier transform and generate a solution based on the spatial-frequency domain data.

In this thesis, some of the current motion estimation algorithms are introduced, discussed, and improved. A comparison between algorithms is established. Chapter II includes the application of the parallel extended Kalman filter. The Extended Kalman Filter is applied to sequential images containing a moving object to estimate object's two-dimensional velocity components and to increase image quality by the reduction of noise

for low signal to noise images. The idea of using an extended Kalman filter (EKF) to enhance the quality of the image frame and provide an estimate of the object velocity was proposed by Burl [Ref. 1]. Chapter III includes the algorithms that utilize the Image Flow Constraint equation which relates the spatial changes in an image frame to the temporal changes between the frames. Several ways for solving this equation are discussed. Chapter IV and V address the Fourier transform methods of motion estimation. These relate phase changes in the Fourier domain representation of image frames to the velocity of objects contained in the image frames. The algorithms discussed in this thesis are simulated under different background and noise conditions to evaluate their performance. Chapter VI includes the simulation results and the final comments.

In this thesis, a moving image model is defined as a series of image frames containing a moving object. An image frame consists of a two-dimensional array where the individual array elements are defined as pixels. Each pixel is given a numerical value based on the intensity of the image at that point.

II. THE EXTENDED KALMAN FILTER

A. INTRODUCTION

An extended Kalman filter (EKF) may be developed for the estimation of both the image and velocity components from a sequence of images in a nearly optimal manner [Ref. 1]. First the EKF requires a dynamic model that describes the evolution of the sequential image frames. A simple dynamic model consists of a shift operator in the spatial domain which describes the motion of the object. Unfortunately the EKF developed from this model requires a prohibitive number of computations. This difficulty is overcome by transforming both image and EKF into the spatial frequency domain. The two-dimensional discrete Fourier transform is used to transform these two-dimensional image frames from the spatial domain to the spatial frequency domain. By transforming the image frame to the spatial frequency domain, the shift operator in the spatial domain becomes a phase shift operator in the spatial frequency domain. The shift of each spatial frequency is given by the scalar product of the velocity and the frequency. The EKF can be applied to each of the spatial frequencies separately. The number of calculations can then be reduced by limiting the number of spatial frequencies that are analyzed.

B. THE MOVING IMAGE MODEL

1. The Moving Image Model in the Spatial Domain

The evolution of successive image frames containing a moving object is modeled by a nonlinear state equation in the spatial domain. The background of the image is assumed to be zero to simplify the initial presentation. A new model for the images with background will be presented later. Also the moving object is assumed to be completely contained in the image. So the state-space equation is defined as,

$$x(k+1) = \begin{bmatrix} f(k+1) \\ v(k+1) \end{bmatrix} = \begin{bmatrix} S(v) & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} f(k) \\ v(k) \end{bmatrix}, \qquad (2.1)$$

$$= a(v)\ x(k),$$

where $x(k)$ is the state vector of the system, $f(k) \in R^{N \times N}$ is the NxN pixel image at time k stacked into a vector of the amplitudes of each pixel, $v \in R^2$ is the velocity vector (consists of a two-element column vector describing the horizontal and vertical velocities of the moving object in terms of pixels per sampling time) and $S(v)$ is a two-dimensional shift operator with magnitude and direction given by v. The vectors f and v combine to form the state vector $x \in R^{N \times N+2}$. Note that object motion is equivalent to a shift operator applied to the entire image since the background is zero. Also the shift operator is a circular shift operator since the object is assumed to remain within the boundary.

Equation 2.1 describes a sequence of identical replications of an object as it moves accross an image frame with constant velocity. A more realistic model would be to include changes in the object due to rotations, change in the aspect angles etc. Additionally, changes in the object's velocity may occur. A new model which includes

these changes may be developed as,

$$x(k+1) = \begin{bmatrix} S(v) & 0 \\ 0 & I \end{bmatrix} x(k) + \begin{bmatrix} \zeta_f(k) \\ \zeta_v(k) \end{bmatrix}, \qquad (2.2)$$

$$= a(v)\, x(k) + \zeta(k),$$

where the plant noise is composed of the image plant noise, ζ_f and the velocity plant noise, ζ_v. The plant noise is assumed to be a zero-mean, white, Gaussian random noise with a covariance matrix defined as,

$$Q_{\zeta\zeta} = E\left[\begin{bmatrix} \zeta_f \\ \zeta_v \end{bmatrix} \begin{bmatrix} \zeta_f^T & \zeta_v^T \end{bmatrix} \right] = \begin{bmatrix} \sigma_f^2 I & 0 \\ 0 & \sigma_v^2 I \end{bmatrix}, \qquad (2.3)$$

where E[•] is the expectation operator and I is the identity matrix. The measurement of the image, $y(k)$, is defined by the corresponding measurement equation,

$$y(k) = f(k)+n(k) = [I\ 0]x(k)+n(k) = C\ x(k)+n(k), \qquad (2.4)$$

where $y \in R^{N\times N}$ is the measured image and $n(k) \in R^{N\times N}$ is the measurement noise which is assumed to be a zero-mean, white, Gaussian random process with a known covariance matrix,

$$R_{nn} = E\left[n(k)\ n^T(k)\right] = \sigma_n^2\, I. \qquad (2.5)$$

2. Derivation of the Shift Operator in the Spatial Frequency Domain

The two-dimensional image frame defined as $f(n_1,n_2)$ where $0 \le n_1 \le N\text{-}1$ and $0 \le n_2 \le N\text{-}1$ is transformed from the spatial domain to the spatial frequency

domain by using the two-dimensional discrete Fourier transform (DFT),

$$F(k_1,k_2) = \sum_{n_1=0}^{N-1} \sum_{n_2=0}^{N-1} f(n_1,n_2)\, e^{-j\frac{2\pi}{N}(n_1k_1+n_2k_2)}, \tag{2.6}$$

where $0 \le k_1 \le N\text{-}1$ and $0 \le k_2 \le N\text{-}1$.

The shift operator in the spatial frequency domain is derived from the circular shift property of the discrete Fourier transform [Ref. 2:p. 67]. The circular shift property is defined as,

$$\begin{aligned} x(\,((n_1-m_1))_N,((n_2-m_2))_N\,) &\leftrightarrow W_N^{n_1k_1}\, W_N^{n_2k_2}\, X(k_1,k_2), \\ &\leftrightarrow D(m_1,m_2)\, X(k_1,k_2), \end{aligned} \tag{2.7}$$

where $((\bullet))$ is the modulus operator, $W_N = e^{-j2\pi/N}$,and $D(m_1,m_2)$ is the transformed shift operator. Equation 2.7 indicates that a circular shift in the spatial domain results in a phase shift in the spatial frequency domain. The original image frame, $x(n_1,n_2)$, is assumed to be a periodic two-dimensional sequence with the fundamental period equal to the frame dimensions. The object moving across the image frame then describes a circular shift [Ref. 2:pp. 61-62]. This implies that, as the object moves off one edge of the screen, it reenters the screen from the opposite side. Real image frames, such as a radar screen or video display, do not require this property because the moving object's size will likely be much smaller than the screen dimensions.

The derivation of $D(m_1,m_2)$ begins by introducing a circular shift into the original image frame, $f(n_1,n_2)$. Equation 2.6 now becomes,

$$F'(k_1,k_2) = \sum_{n_1=0}^{N-1} \sum_{n_2=0}^{N-1} f(n_1-m_1,n_2-m_2)\, e^{-j2\pi\left(\frac{n_1k_1}{N}+\frac{n_2k_2}{N}\right)}, \tag{2.8}$$

where $F'(k_1,k_2)$ is the circular-shifted, transformed image frame. The amount of pixel movement is described by m_1 for the horizontal shift and m_2 for the vertical shift. Letting $l_1=n_1-m_1$ and $l_2=n_2-m_2$ Equation 2.8 becomes,

$$F'(k_1,k_2) = \sum_{l_1=-m_1}^{N-m_1-1} \sum_{l_2=-m_2}^{N-m_2-1} f(l_1,l_2) e^{-j2\pi\left(\frac{(l_1+m_1)k_1}{N}+\frac{(l_2+m_2)k_2}{N}\right)}. \tag{2.9}$$

Rearranging terms, Equation 2.9 reduces to,

$$\begin{aligned} F'(k_1,k_2) &= e^{-j2\pi\left(\frac{m_1k_1}{N}+\frac{m_2k_2}{N}\right)} F(k_1,k_2), \\ &= W_N^{m_1k_1}\, W_N^{m_2k_2}\, F(k_1,k_2), \\ &= D(m_1,m_2)\, F(k_1,k_2), \end{aligned} \tag{2.10}$$

where $F(k_1,k_2)$ is the transformed image frame without the circular shift. This proves that, when an object moves across an image frame in the spatial domain, the motion is described by a phase shift in the spatial frequency domain, thereby agreeing with the circular shift property (Equation 2.8).

3. The Moving Image Model In The Spatial Frequency Domain

The spatial domain model described by Equation 2.3 can be transformed to the spatial frequency domain. The transformation is accomplished by taking the discrete Fourier transform of the image portion of these equations which gives,

$$\begin{bmatrix} F(k+1) \\ v(k+1) \end{bmatrix} = \begin{bmatrix} D(v) & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} F(k) \\ v(k) \end{bmatrix} + \begin{bmatrix} z_F(k) \\ \zeta_v(k) \end{bmatrix}$$

$$X(k+1) = A(v)\ X(k) + Z(k) \tag{2.11}$$

$$= a(X(k),Z(k))$$

where,

$$F(k) = \mathcal{F}\{f(k)\} \in \mathbb{C}^{N^2}, \tag{2.12}$$

$$z_F(k) = \mathcal{F}\{\zeta_f(k)\} \in \mathbb{C}^{N^2}, \tag{2.13}$$

$$D(v) = diagonal\left[\ e^{j\omega_i v/N}\ \right]. \tag{2.14}$$

and $a(\bullet,\bullet)$ *is a non-linear operator.* The vector, $f(k)$ and its corresponding plant noise, $\zeta_f(k)$ are transformed to the spatial frequency domain, where $\mathcal{F}\{\bullet\}$ is the two-dimensional discrete Fourier transform operator (Note that the index k refers to the sample at time k). The transformed shift operator, $D(v)$ is a diagonal matrix where the diagonal terms $D_i(v)$ are the transformed shift operators for specified spatial frequencies so that the corresponding elements ω_i of spatial frequency vector ω is defined as,

$$\omega_i = \begin{bmatrix} \frac{2\pi k_1 i}{N} \\ \frac{2\pi k_2 i}{N} \end{bmatrix}. \tag{2.15}$$

The image plant noise remains a zero-mean, white Gaussian random process since it is generated by a linear operation:

$$Q_{zz} = E[zz^T] = \begin{bmatrix} \sigma_F^2 I & 0 \\ 0 & \sigma_v^2 I \end{bmatrix}. \tag{2.16}$$

Additionally, note that only half of the spatial frequencies need to be stored in F(k) due to the conjugate symmetry property of discrete Fourier transform of a real image [Ref. 3:p. 45].

The corresponding measurement equation in the spatial frequency domain is:

$$\begin{aligned} Y(k) &= F(k) + N(k), \\ &= [I\ 0]\ X(k) + N(k), \\ &= C\ X(k) + N(k), \end{aligned} \tag{2.17}$$

where $Y(k) = \mathcal{F}(y(k))$ and $N(k) = \mathcal{F}(v(k))$, and N is a zero-mean gaussian with known covariance matrix,

$$R_{NN} = E[N(k)\ N^T(k)] = \sigma_n^2 I. \tag{2.18}$$

In practice, the image is measured and then the Fourier transform is performed. But for simplicity, we will assume that the Fourier transform of the image is measured. The state equations given by Equations 2.11 through 2.18 are the basis for implementing the EKF in the spatial frequency domain. The Equations 2.11 through 2.18 define a nonlinear state space model of a sequence containing a moving object. The state of this system consist of both the image (in the spatial frequency domain) and the velocity of the moving object. The model is driven by white noise. Measurements consist of the image in the spatial frequency domain corrupted by additive noise. The extended Kalman filter has found wide application as a state estimator for systems of the form described above.

C. THE EXTENDED KALMAN FILTER

1. The Extended Kalman Filter

The extended Kalman filter is a nonlinear, recursive filter that can be employed to generate estimates of a system whose evolution is governed by a nonlinear state equation. The extended Kalman filter equations for the moving image model are:

State Prediction:

$$\hat{X}(k+1|k) = a(\hat{X}(k|k),0), \tag{2.19}$$

$$\hat{Y}(k+1|k) = C\hat{X}(k+1|k), \tag{2.20}$$

Covariance Prediction:

$$P(k+1|k) = A(\hat{X}(k|k))P(k|k)A^T(\hat{X}(k|k)) + Q_{zz}, \tag{2.21}$$

Kalman Gain:

$$G(k+1) = P(k+1|k)C^T[CP(k+1|k)C^T + R_{NN}]^{-1}, \tag{2.22}$$

State Correction:

$$\hat{X}(k+1|k+1) = \hat{X}(k+1|k) + G(k+1)[Y(k+1) - \hat{Y}(k+1|k)], \tag{2.23}$$

Covariance Correction:

$$P(k+1|k+1) = [I - G(k+1)C]P(k+1|k). \tag{2.24}$$

The ^ denotes an estimate and the notation $k+1|k$ is read as: at time k+1 given data through time k. These equations are a set of recursive equations that can be used for estimating the state X(k).

2. The Modified Extended Kalman Filter

As mentioned earlier, the diagonal properties of the transformed shift operator, $D(v)$, and the covariance matrices are the basis for the parallel structure shown in Figure 2.1. This parallel structure is referred to as the modified extended Kalman filter. Reference 1 outlines how the EKF converges to the MEKF as the velocity estimate approaches the actual velocity. Practical implementation of the MEKF suggests that each filter in the parallel bank be dedicated to a specific spatial frequency. Each individual filter is referred to as a single frequency extended Kalman filter (SFEKF). The state vector for a specific spatial frequency is defined as,

$$X_i(k) = \begin{bmatrix} X_{i1}(k) \\ X_{i2}(k) \\ X_{i3}(k) \end{bmatrix} = \begin{bmatrix} Re(F_i(k)) \\ Im(F_i(k)) \\ \omega_i^T v \end{bmatrix}, \tag{2.25}$$

where the first two states, $X_1(k)$ and $X_2(k)$ are the real and imaginary parts of Fourier coefficients associated with a specific spatial frequency and the third state, $X_3(k)$, is the product of the spatial frequency vector and the velocity vector. $X_{i3}(k)$ is called the velocity-frequency product for each frequency, Equation 2.11 becomes,

$$\begin{bmatrix} Re(F_i(k+1)) \\ Im(F_i(k+1)) \\ \omega_i^T v \end{bmatrix} = \begin{bmatrix} \cos(\omega_i^T v) & \sin(\omega_i^T v) & 0 \\ -\sin(\omega_i^T v) & \cos(\omega_i^T v) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Re(F_i(k)) \\ Im(F_i(k)) \\ \omega_i^T v \end{bmatrix}, \tag{2.26}$$

where v contains the two velocity components as,

$$\nu = \begin{bmatrix} \nu_1 \\ \nu_2 \end{bmatrix}. \tag{2.27}$$

The non-linear state matrix of Equation 2.26 can be linearized around the current estimate of the state by using the Jacobian operator,

$$A(\hat{X}(k|k)) = \left.\frac{\partial a}{\partial X}\right|_{X=\hat{X}(k|k)}. \tag{2.28}$$

The linearized state equation for a single frequency is,

$$\hat{X}_i(k+1) = A_i(\hat{X}_i(k))\ \hat{X}_i(k) + Z_i(k), \tag{2.29}$$

where,

$$A_i(\hat{X}_i(k)) =$$

$$\begin{bmatrix} \cos(\hat{X}_{i3}(k)) & \sin(\hat{X}_{i3}(k)) & [-\hat{X}_{i1}(k)\sin(\hat{X}_{i3}(k))+\hat{X}_{i2}(k)\cos(\hat{X}_{i3}(k))] \\ -\sin(\hat{X}_{i3}(k)) & \cos(\hat{X}_{i3}(k)) & [-\hat{X}_{i1}(k)\cos(\hat{X}_{i3}(k))-\hat{X}_{i2}(k)\sin(\hat{X}_{i3}(K))] \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.30}$$

The single frequency model (Equation 2.29), the measurement equation (Equation 2.20), the linearized state matrix (Equation 2.30) combine with Kalman filter equations to fully specify the single frequency EKF's. Application of these equations yields estimates of the real and imaginary components of the Fourier coefficients and the frequency-velocity products.

D. VELOCITY ESTIMATION

The single frequency EKF's each yield an estimated frequency-velocity product. These estimates can be combined to yield an estimate of the velocity of the moving object. This final estimate is computed by using a weighted least-squares algorithm. Unfortunately this estimation algorithm is complicated by an ambiguity of multiplies of 2π in the frequency-velocity product estimates. This ambiguity is distinct from the problem of phase unwrapping since the single frequency EKF's are free to converge to any of the possible multiplies and do not have the rigid structure to the phase unwrapped signal [Ref. 1]. By including these ambiguity terms the frequency-velocity estimates can be modelled by,

$$\tilde{X}_3(k|k) = \Omega v + w + 2\pi m, \tag{2.31}$$

where,

$$\Omega = \frac{1}{N}[\omega_1, \omega_2, \ldots, \omega_{N^2}], \tag{2.32}$$

$$w = [w_1, w_2, \ldots, w_{N^2}], \tag{2.33}$$

and

$$m = [m_1, m_2, \ldots, m_{N^2}]. \tag{2.34}$$

The term w is a zero-mean, Gaussian random vector with the estimated covariance,

$$\Sigma = E[ww^T] = diag[P_{133}, P_{233}, \ldots, P_{N^2 33}] \tag{2.35}$$

where P_{ii3} is the *3,3* component of the *i*th estimation error covariance matrix computed

by each of the SFEKF's. As stated before, the frequency-velocity product enters the state equation only as a phase term and is ambiguous with respect to a shift of a multiple of 2π. The term m is an unknown integer vector which models this ambiguity.

The estimates from Equation 2.31 can be used to estimate the velocity by finding the real vector v and integer vector m, that minimizes the weighted sum of the squared errors,

$$J = \left(\hat{X}_3(k|k) - \Omega v - 2\pi m\right)^T \Sigma^{-1} \left(\hat{X}_3(k|k) - \Omega v - 2\pi m\right). \tag{2.36}$$

The vectors v and m that minimize Equation 2.36 can be found by performing a search over all possible values of m. For each value of m, finding the value of v that minimizes Equation 2.36 is equivalent to the classical Weighted Least Squares problem. A new data vector may be defined as,

$$\tilde{X} = \hat{X}_3 (k|k) - 2\pi m, \tag{2.37}$$

and the sum of the squared error is,

$$J = (\tilde{X} - \Omega v)^T \Sigma^{-1} (\tilde{X} - \Omega v). \tag{2.38}$$

Constraining the velocity estimate to be a linear function of the data yields,

$$\hat{v} = G_v \tilde{X}_3(k|k), \tag{2.39}$$

where,

$$G_v = (\Omega^T \Sigma^{-1} \Omega)^{-1} \Omega^T \Sigma^{-1}. \tag{2.40}$$

Substituting Equation 2.39 into Equation 2.36 yields the sum of the error as a function

of m,

$$J(m) = (\hat{X}(k|k) - 2\pi m)^T \Sigma^{-1} (I - \Omega G_v)(\hat{X}_3(k|k) - 2\pi m). \quad (2.41)$$

Using Equation 2.41, m can be estimated using a search of all possible vectors m to find the one that minimizes $J(m)$.

A simplification to this search for the optimal vector m is developed by Burl [Ref. 1:p. 12]. The values of the m_i's will be zero for low spatial frequencies. So by using this subset of frequency components, the linear estimator of Equation 2.39 yields v_{low}. This velocity estimate is used to estimate the m_i's that do not meet the condition $m_i = 0$ by,

$$\hat{m}_i = Round\left(\frac{1}{2\pi}[\hat{X}_{i3}(k|k) - \omega_i^T \hat{v}_{low}]\right), \quad (2.42)$$

where $Round(\bullet)$ equals the integer nearest to $\bullet$. This estimate, in turn, is used to estimate the velocity using Equation 2.37 and 2.39. The estimated values of m are included in the frequency-velocity product estimates that are fed back to single frequency EKF's. Therefore, once the filter has converged the values of m will all be zero.

E. IMAGES WITH NONZERO BACKGROUND

The background in most image sequences will be non-zero. A method of modifying the MEKF for use in this case is proposed by Burl [Ref. 1]. This method proposes that 2 more states are added to the single frequency EKF's. The additional states are defined as the real and imaginary portions of the background in the spatial frequency domain and,

$$X_i(k) = \begin{bmatrix} X_{i1}(k) \\ X_{i2}(k) \\ X_{i3}(k) \\ X_{i4}(k) \\ X_{i5}(k) \end{bmatrix} = \begin{bmatrix} Re(F_i(k)) \\ Im(F_i(k)) \\ Re(F_{si}(k)) \\ Im(F_{si}(k)) \\ \omega_i^T v(k) \end{bmatrix}, \tag{2.43}$$

where $F_{si}(k)$ is the Fourier transform of the stationary background. The state transition matrix for these new states are simply the identity matrix since the background is considered to be constant. The state matrix of Equation 2.30 then becomes,

$$A_i(\hat{X}_i(k)) = \begin{bmatrix} \cos(\hat{X}_{i5}(k)) & \sin(\hat{X}_{i5}(k)) & 0 & 0 & [-\hat{X}_{i1}(k)\sin(\hat{X}_{i5}(k)) + \hat{X}_{i2}(k)\cos(\hat{X}_{i5}(k))] \\ -\sin(\hat{X}_{i5}(k)) & \cos(\hat{X}_{i5}(k)) & 0 & 0 & [-\hat{X}_{i1}(k)\cos(\hat{X}_{i5}(k)) - \hat{X}_{i2}(k)\sin(\hat{X}_{i5}(K))] \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.44}$$

In this case the measurement is modeled as the sum of the moving object state, the stationary background state, and the additive noise.

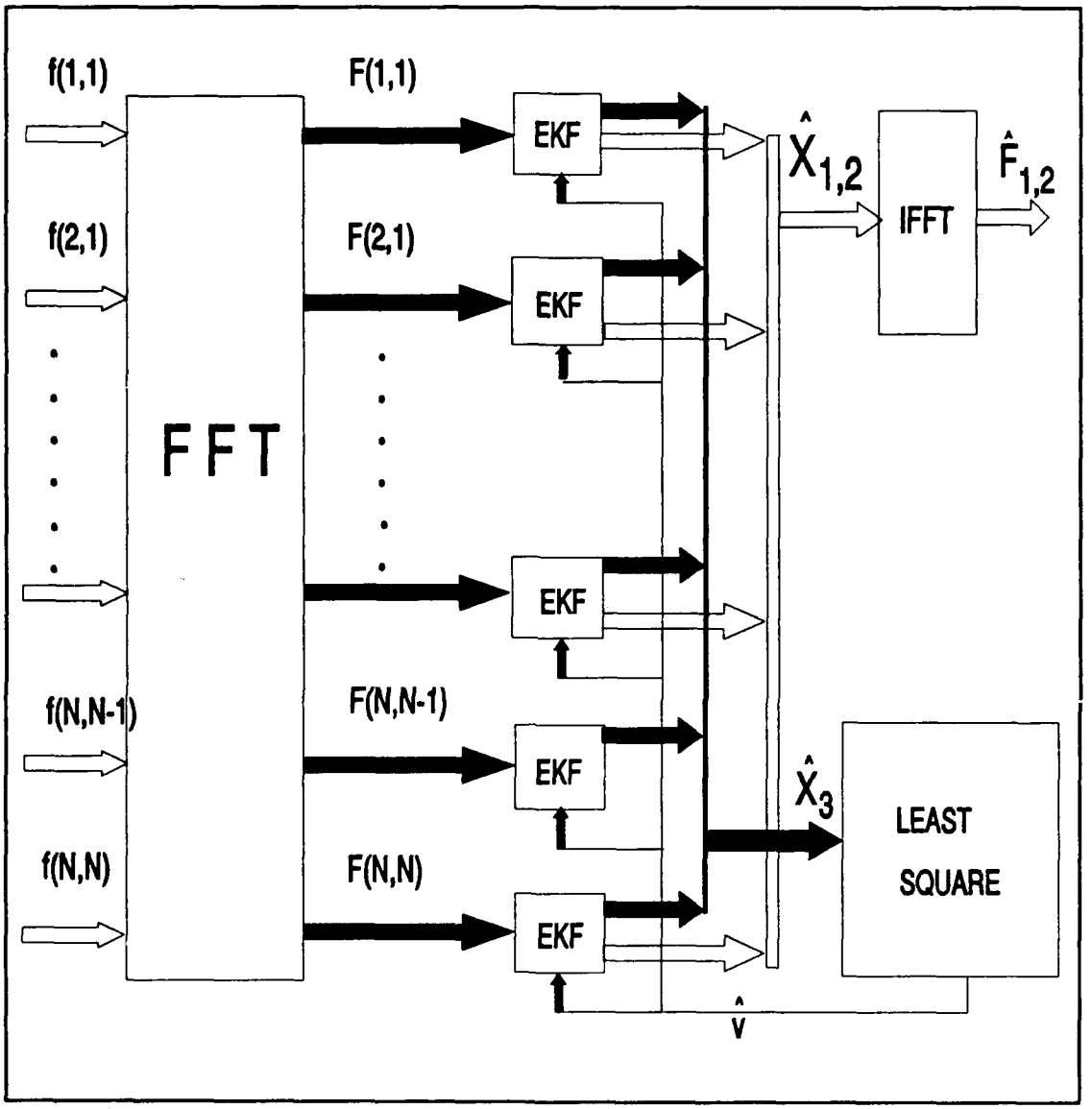


Figure 2.1 The Modified Extended Kalman Filter.

III. SPATIO - TEMPORAL GRADIENT APPROACH

A. THE IMAGE FLOW CONSTRAINT EQUATION

1. The Image Flow Constraint Equation

This method for computing the velocity field (optical flow) employs the first-order spatial and temporal differentials of a time varying image to estimate the component of motion at each point. We can derive an algorithm that relates the change in image brightness at a point to the motion of the brightness pattern. Let the image brightness at a point (x,y) in the image plane at time t be denoted by $E(x,y,t)$. Now consider that the pattern moves. The brightness of a particular point in the pattern is constant, so that,

$$\frac{dE}{dt} = 0. \tag{3.1}$$

Using the chain rule for differentiation we see that,

$$\frac{\partial E}{\partial x}\frac{dx}{dt} + \frac{\partial E}{\partial y}\frac{dy}{dt} + \frac{\partial E}{\partial t} = 0. \tag{3.2}$$

(A more detailed derivation with Taylor series expansion is included in Appendix A). This equation can simply be written as,

$$E_x\, u + E_y\, v + E_t = 0, \tag{3.3}$$

where,

$$u = \frac{dx}{dt}, \qquad v = \frac{dy}{dt},$$

$$E_x = \frac{\partial E}{\partial x}, \quad E_y = \frac{\partial E}{\partial y}, \quad E_t = \frac{\partial E}{\partial t}. \tag{3.4}$$

The partial derivatives E_x, E_y, E_t are estimated from the image intensity values and will be formulated later. The constraint given by Equation 3.2 on the local flow velocity is illustrated in Figure 1 [Ref. 3:pp. 20-46].

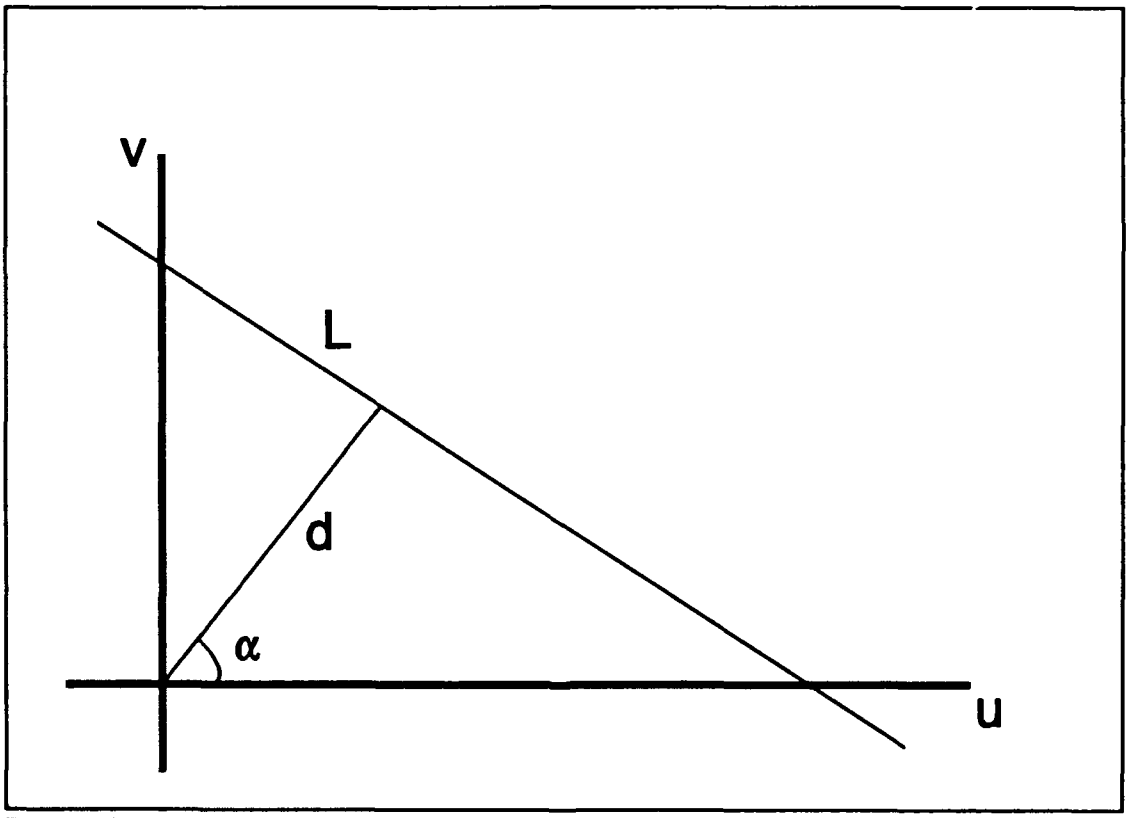


Figure 3.1. The locus of points satisfied by the image flow constraint equation defines a line in the velocity space.

This constraint is often called as *the image flow constraint equation* since it expresses a constraint on the components u and v of the image flow. As seen in Figure 3.1, values of (u,v) satisfying the constraint equation lie on a straight line in velocity space. Two conditions are required to ensure the validity of the image flow constraint equation; (1) the change in image irradiance at each point must be entirely due to the motion of the image pattern, as opposed to changes in the pattern due to reflectance effects, and (2) the image should be smooth except at a finite number of discontinuities [Ref. 4:pp. 185-189]. Surface based smoothing may be used to improve the velocity estimates for images with discontinuities. These conditions are probably too restrictive to allow image flow to be useful, but the image flow equation is obeyed in practice with sufficient accuracy for tasks such as segmentation.

The image flow can not be computed at a point in the image independently of neighboring points without introducing additional constraints. This limitation occurs because the velocity field at each point has two components while the change in image irradiance at a point due to motion yields only one constraint. Estimation approaches for this problem, in one manner or another introduce additional constraints that allow mathematical solutions. One solution to this dilemma is to assume that in the local image region there are two values of $E(x,y,t)$ that have the same image motion, i.e. adjacent pixels exhibit similar image motion. Defining these two points as $E_1 = E(x_1,y_1,t)$ and $E_2 = E(x_2,y_2,t)$, we can write the set of two equations,

$$\frac{\partial E_1}{\partial x}u + \frac{\partial E_1}{\partial y}v + \frac{\partial E_1}{\partial t} = 0,$$
$$\frac{\partial E_2}{\partial x}u + \frac{\partial E_2}{\partial y}v + \frac{\partial E_2}{\partial t} = 0, \tag{3.5}$$

or the matrix equation,

$$\underline{E_t} = S_{xy}\,\underline{u}, \tag{3.6}$$

where $\underline{u} = [u\ v]^T$, and the matrix S is formed with the appropriate values from the equations above. Inverting S yields $[u\ v]^T$ [Ref. 5:pp. 230-238]. However, any errors in the spatial and temporal derivative estimates may have severe effect on the velocity estimates. Furthermore there is no guarantee that S is invertible. The local motion homogeneity, although perhaps valid for image points corresponding to object points that are close together on the moving object, is not a valid assumption when adjacent image points correspond to the motion of independently moving and occluding objects.

An alternative approach is to explicitly formulate a *local smoothness constraint* on the velocity vector, that is to assume that objects of small size are undergoing rigid motion and deformation. In this case neighboring points on the objects have similar velocities and the velocity field of the brightness patterns in the image varies smoothly almost everywhere. Discontinuities in flow can be expected at the object boundaries and also where one object occludes another. This may be formulated mathematically as the additional constraint which is to minimize the sum of the squares of the Laplacians of the x and y components of the flow. The Laplacians of u and v are defined as,

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \quad \textit{and} \quad \nabla^2 v = \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}. \tag{3.7}$$

Now we can introduce the optimization problem. The optimization measure consist of two terms: a penalty on the deviation of estimated velocity field from the image flow constraint equation and a penalty on the departure of velocity components from smoothness (the smoothness penalty was the sum of the squares of the magnitude of the gradient of the image flow velocity). These constraints may be written as,

$$\begin{aligned} &\textit{Minimize,} \qquad \varepsilon_1 = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \\ &\textit{subject to,} \qquad \varepsilon_2 = E_x u + E_y v + E_t = 0 . \end{aligned} \tag{3.8}$$

The total error to be minimized becomes,

$$\varepsilon_T = \varepsilon_1^2 + \lambda \varepsilon_2^2. \tag{3.9}$$

2. Estimating the Partial Derivatives and the Laplacian of Flow Velocities

We must estimate the derivatives of the brightness from the discrete set of image brightness measurements available. The consistency of partial derivatives are very important. That is they should refer to the same point in the image at the same time. Although there are many formulas for approximate differentiation, an optimum method given by Schunk [Ref. 4:p. 190] can be used, which gives an estimate of E_x, E_y, E_t at a point in the center of a cube formed by eight measurements. The relationship in space and time between these measurements is shown in Figure 3.2. Each of these estimates is the average of four first differences taken over adjacent measurements in the cube.

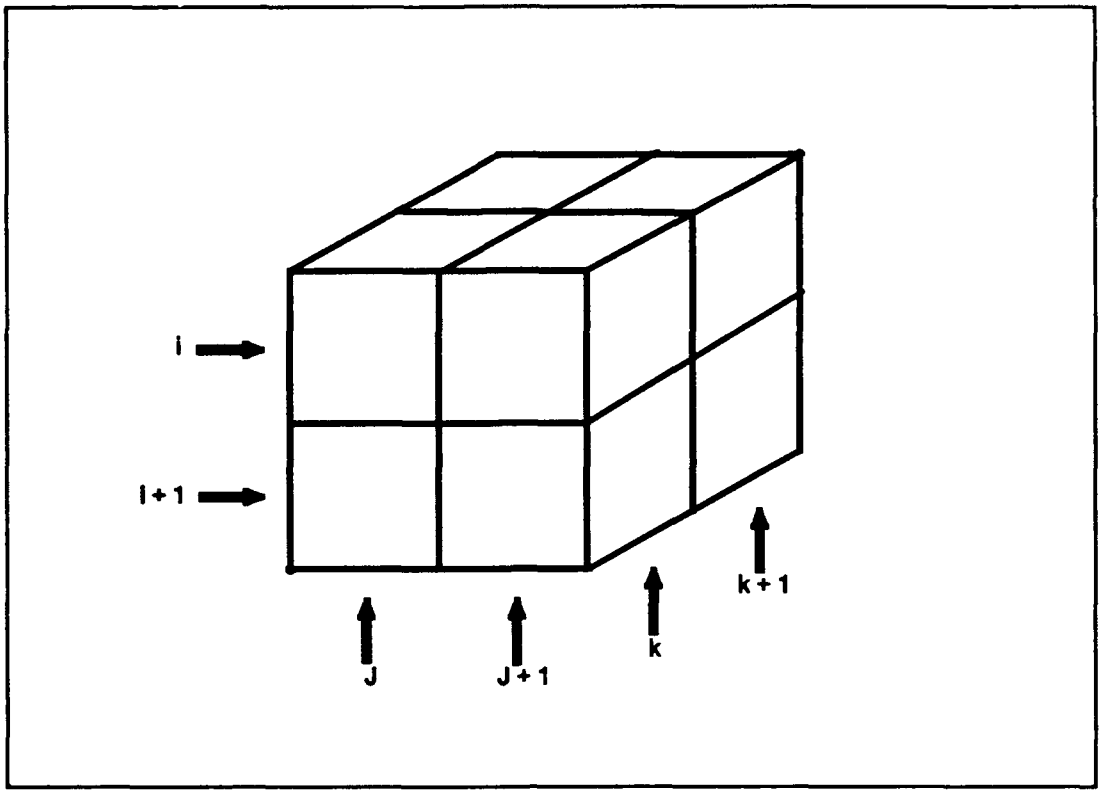


Figure 3.2. The three partial derivatives of image brightness are estimated from the average of first differences along four parallel edges of the cube.

$$
\begin{aligned}
E_x &\approx \frac{1}{4}\{E_{i,j+1,k}-E_{i,j,k}+E_{i+1,j+1,k}-E_{i+1,j,k} \\
&\quad +E_{i,j+1,k+1}-E_{i,j,k+1}+E_{i+1,j+1,k+1}-E_{i+1,j,k+1}\}, \\
E_y &\approx \frac{1}{4}\{E_{i+1,j,k}-E_{i,j,k}+E_{i+1,j+1,k}-E_{i,j+1,k} \\
&\quad +E_{i+1,j,k+1}-E_{i,j,k+1}+E_{i+1,j+1,k+1}-E_{i,j+1,k+1}\}, \\
E_t &\approx \frac{1}{4}\{E_{i,j,k+1}-E_{i,j,k}+E_{i,j+1,k+1}-E_{i,j+1,k} \\
&\quad +E_{i+1,j,k+1}-E_{i+1,j,k}+E_{i+1,j+1,k+1}-E_{i+1,j+1,k}\}.
\end{aligned}
\tag{3.10}
$$

We also need to approximate Laplacians of u and v. One convenient approximation takes the following form,

$$\nabla^2 u \approx (\bar{u} - u) \quad and \quad \nabla^2 v \approx (\bar{v} - v), \tag{3.11}$$

where the local averages $\bar{u}$ and $\bar{v}$ are defined as follows,

$$\begin{aligned}
\bar{u}_{i,j,k} &= \frac{1}{6}\{u_{i+1,j,k}+u_{i-1,j,k}+u_{i,j+1,k}+u_{i,j-1,k}\} \\
&+\frac{1}{12}\{u_{i+1,j+1,k}+u_{i+1,j-1,k}+u_{i-1,j+1,k}+u_{i-1,j-1,k}\}, \\
\bar{v}_{i,j,k} &= \frac{1}{6}\{v_{i+1,j,k}+v_{i-1,j,k}+v_{i,j+1,k}+v_{i,j-1,k}\} \\
&+\frac{1}{12}\{v_{i+1,j+1,k}+v_{i+1,j-1,k}+v_{i-1,j+1,k}+v_{i-1,j-1,k}\}.
\end{aligned} \tag{3.12}$$

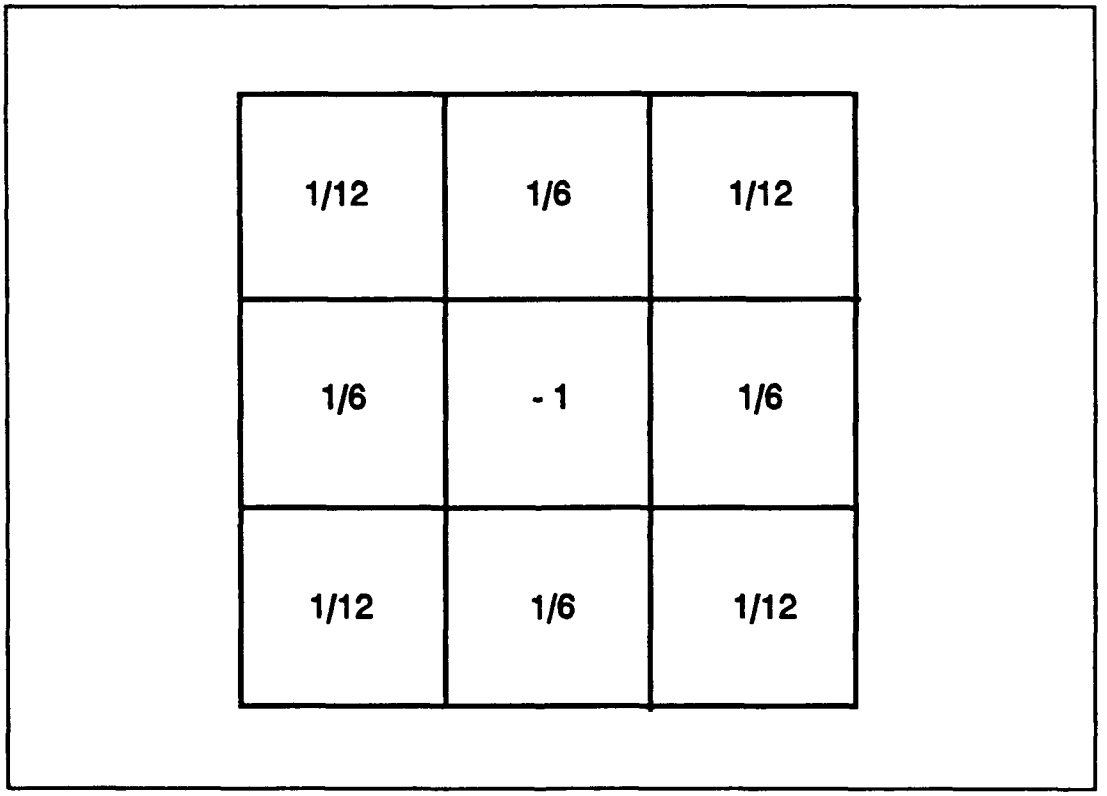


Figure 3.3 The Laplacian is estimated from a weighted average of the values at neighboring points.

3. Minimization

The minimization of total error ε_T is a *constrained minimization process* which has been formulated by Lagrange multipliers in Equation 3.9 (there are several other ways). The solution to the Lagrange formulation is given by finding values of $\underline{u} = [u,v]^T$ that satisfy,

$$\begin{aligned} \frac{\partial \varepsilon_T}{\partial \underline{u}} &= 0, \\ \frac{\partial \varepsilon_T}{\partial \lambda} &= 0. \end{aligned} \tag{3.13}$$

Using the calculus of variations [Ref. 6:pp. 107-178] yields the solution as,

$$\begin{aligned} E_x^2\, u + E_x\, E_y\, v &= \lambda\, \nabla^2 u - E_x\, E_t, \\ E_x\, E_y\, u + E_y^2\, v &= \lambda\, \nabla^2 v - E_y\, E_t. \end{aligned} \tag{3.14}$$

Using the approximation to the Laplacian introduced previously gives the folowing equations,

$$\begin{aligned} (\lambda + E_x^2)u + E_x\, E_y\, v &= (\lambda\, \bar{u} - E_x\, E_t), \\ (E_x\, E_y\, u + (\lambda + E_y^2)v &= (\lambda\, \bar{v} - E_y\, E_t). \end{aligned} \tag{3.15}$$

Solving for u and v gives,

$$\begin{aligned} u &= \bar{u} - (E_x^2\, \bar{u} + E_x\, E_y\, \bar{v} + E_x\, E_t)/(\lambda + E_x^2 + E_y^2) \\ v &= \bar{v} - (E_x\, E_y\, \bar{u} + E_y^2\, \bar{v} + E_y\, E_t)/(\lambda + E_x^2 + E_y^2) \end{aligned} \tag{3.16}$$

These results yield two solutions for λ such that,

$$\begin{aligned} \lambda &= -[E_x^2\, u + E_x\, E_y\, v + E_x\, E_t]/(u - \bar{u}), \\ &= -[E_x\, E_y\, u + E_y^2\, v + E_y\, E_t]/(v - \bar{v}). \end{aligned} \tag{3.17}$$

Notice that these equations constrain u and v but return no new information. They also yield,

$$\varepsilon_2 = 0.$$

Now we have a set of two equations for each point in the image. It would be very costly to solve these equations simultaneously (also with the Lagrange multipliers). Instead an iterative method can be developed. We can compute a new set of velocity estimates (u^{n+1}, v^{n+1}) from the estimated derivatives and the average of previous velocity estimates (u^n, v^n) by,

$$\begin{aligned} u^{n+1} &= \bar{u}^n - E_x[E_x\, \bar{u}^n + E_y\, \bar{v}^n + E_t]/(\lambda + E_x^2 + E_y^2), \\ v^{n+1} &= \bar{v}^n - E_y[E_x\, \bar{u}^n + E_y\, \bar{v}^n + E_t]/(\lambda + E_x^2 + E_y^2). \end{aligned} \tag{3.18}$$

The algorithm stops when convergence is obtained: that is when,

$$\begin{aligned} u^n(x,y,t) - u^{n-1}(x,y,t) &\le e_1, \\ v^n(x,y,t) - v^{n-1}(x,y,t) &\le e_2. \end{aligned} \tag{3.19}$$

In parts of the image where the brightness gradient is zero, the velocity estimates will simply be the averages of the neighboring velocity estimates. There is no local information to constraint the apparent velocity of motion of the brightness pattern in these areas. Eventually the values in these areas will propagate inward. If the velocities on the border of a region are all equal to the same value, then points in the

region will be assigned that value too (after a sufficient number of iterations). If the values on the border are not the same, it is a little more difficult to predict what will happen. In all cases the values filled in will correspond to the solution of the Laplace equation for a given boundary condition.

The best result in most cases is provided if the algorithm is performed a small number of times per frame of a test sequence. In the first few frames an image flow velocity will be estimated within the boundaries of moving object. The velocity field outside the object area will be zero (or very small). As the object moves, the region of nonzero velocity field does not move along with the moving object, since there is nothing in the algorithm that forces the velocity field to be extrapolated in the direction of motion. Since the location of the region of nonzero velocity field is not forced to move along with the object, the translating object moves out from under the region of nonzero velocity vectors. New nonzero velocity vectors are formed at the new position of the object. The algorithm does not eliminate the old region of velocity that trails behind the moving object. This creates a *'comet tail'* behind the object. In regions where the gradient is zero, the algorithm degenerates into the Laplace equation which smooths the velocity field. The region of nonzero velocity vectors that trails behind the moving object is also smoothed into a consistent (though obviously wrong) velocity field which persist long after the object moves beyond the boundaries of the image. As mentioned before, the algorithm cannot work in cases where there are motion boundaries in the velocity field, or more than one moving object exist int he scene, since the smoothness constraint leads to iterative equations that blur abrupt changes in the velocity field.

One may use a smoothing algorithm for improving the velocity field estimation. The smoothing algorithm constructs a smooth estimate of the velocity field by approximating a surface between step discontinuities. Another algorithm called *Constraint Line Clustering* that utilizes the polar form of the image flow constraint equation to estimate the image flow velocity field when there are discontinuities is described by Schunk [Ref. 7:pp. 1010-1027].

B. CONSTRAINT LINE CLUSTERING

1. The Polar Form of the Constraint Equation

The image flow constraint equation defines a line in velocity space as shown in Figure 3.1. The line in velocity space that is the locus of possible velocities specified by the image flow constraint equation is called *the constraint line*. The constraint line may be uniquely defined by d, the distance of the line from the origin along the perpendicular bisector and the angle α with respect to the u axis. The displacement and the angle of the constraint line are given by,

$$d = \frac{|E_t|}{\sqrt[2]{E_x^2 + E_y^2}}, \tag{3.20}$$

$$\alpha = \begin{cases} \arctan(E_x, E_y) & \text{if } E_t \geq 0; \\ \arctan(-E_x, -E_y) & \text{otherwise.} \end{cases} \tag{3.21}$$

To derive the constraint equation in polar coordinates, we first need to note that the image flow equation contains the dot product of the gradient of the image irradiance with

the velocity vector,

$$E_x u + E_y v + E_t = \nabla E \cdot (u, v) + E_t. \tag{3.22}$$

If we define ρ as the speed of motion and β as the direction of motion contained in the constraint equation, then the dot product in Equation 3.22 becomes,

$$\rho \, |\nabla E| \cos(\alpha - \beta). \tag{3.23}$$

Dividing through by the magnitude of the gradient yields the polar form of the image flow constraint equation,

$$d = \rho \cos(\alpha - \beta). \tag{3.24}$$

Note that β is constrained to be between $\alpha-\pi/2$ and $\alpha+\pi/2$. Since the displacement d of the constraint line from the origin must be nonnegative, the orientation α is reflected when $E_t > 0$. The displacement d is the projection of the motion vector onto the line of the gradient of image irradiance and is independent of the magnitude or polarity of the gradient.

The polar form of the image flow constraint equation can be used in the analysis of algorithms for image flow estimation with motion boundaries since the polar form will not contain δ-functions at step discontinuities. The polar form will not contain δ-functions since the displacement d is the ratio of the magnitude of the change in time of the image irradiance to the magnitude of the gradient of image irradiance and this ratio remains finite in the limit as the image irradiance becomes an ideal step discontinuity.

The constraints are sufficient to allow the sampled image sequence to

accurately portray the local motion data between motion boundaries, but are not sufficient to capture the motion information at motion boundaries. The motion constraints are extremely sensitive to the effects of occlusion boundaries and are usually in severe error. No practical sampling period can provide useful motion information along a motion boundary. An image flow estimation algorithm must be able to work in situations where there are motion boundaries in the image. So the algorithm called constraint line clustering [Ref. 7:pp. 1010-1027] that estimates the image flow field when the image irradiance pattern or the velocity field contains discontinuities is presented in the next section.

2. Constraint Line Clustering

The constraint line clustering algorithm uses the polar form of the image flow constraint equation given by,

$$d = \rho \cos(\alpha - \beta), \tag{3.24}$$

where $\rho(x,y)$ and $\beta(x,y)$ are the speed and direction of motion, respectively. The velocity vector (ρ,β) for the motion at any point in the image must lie along the line in velocity space defined by the image flow constraint equation shown in Figure 3.1. The constraint line is uniquely defined by the displacement d of the constraint line from the origin and the orientation α of the constraint line. The constraint line displacement has the dimensions of speed; it is the minimum speed consistent with the motion constraint.

Assume that the image intensity $E(x,y,t)$ already incorporates any spatial or temporal filtering performed on the image. For example, the image could be smoothed

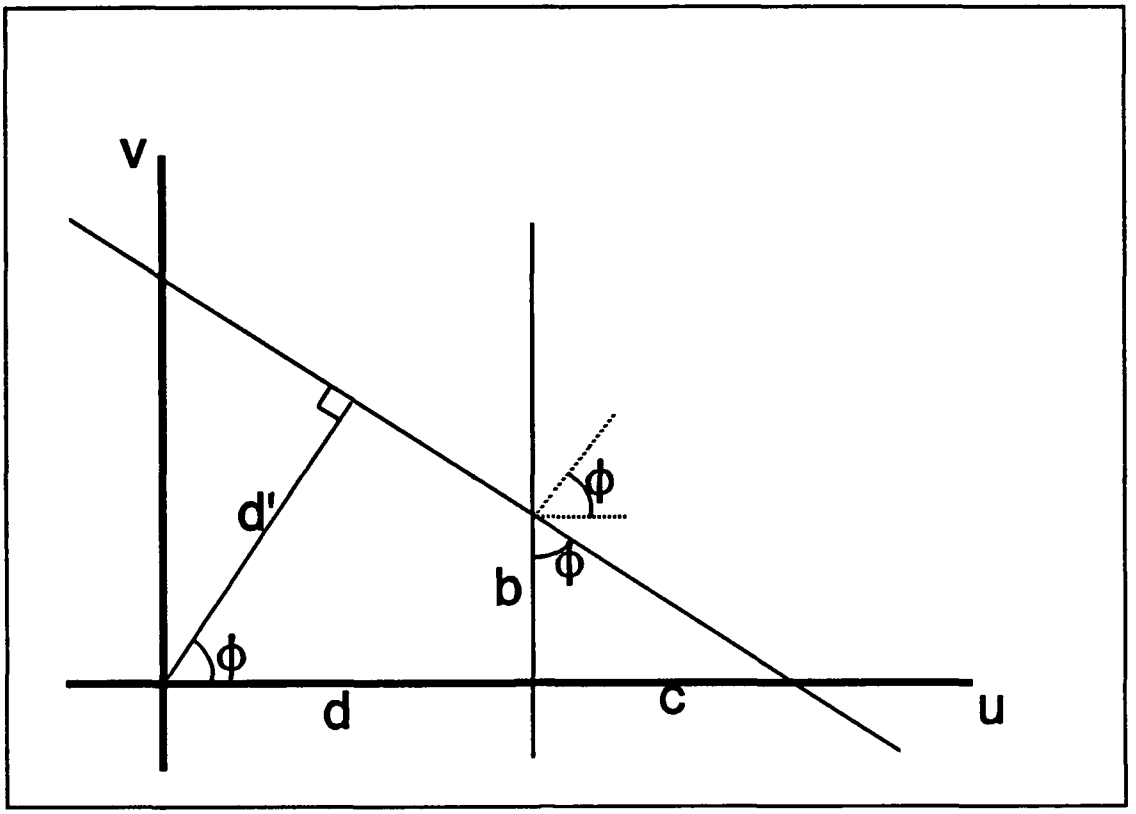


Figure 3.4 The position of the intersection of two constraint lines along one of the constraint lines.

with a Gaussian filter. The image is sampled in space and time to produce an image sequence $\dot{E}(x_i,y_j,t_k)$. The d and α intrinsic image arrays $d(x_i,y_j)$ and $\alpha(x_i,y_j)$ are computed from the image sequence using the formulas in Equations 3.20 and 3.21. The partial derivatives are computed by the differences over a cube in space and time by equations given by Equation 3.10. The image flow estimation problem is to compute the intrinsic image arrays $\rho(x_i,y_j)$ and $\beta(x_i,y_j)$ for the speed and direction of motion from the motion measurements in the d and α arrays. The constraint line clustering algorithm uses a cluster analysis in one dimension to extract the motion estimate from contradictory data.

Suppose that for each d and α measurement, a set of measurements $\{ d_i, \alpha_i \}$ is taken from some spatial neighborhood of the point that generated d and α. Compute the set of intersections of each of the neighboring measurements d_i and α_i with the given d and α measurement. All of these intersections lie along the line defined by d and α. Any constraint lines in $\{ d_i, \alpha_i \}$ that are part of the same region of motion as d and α will tend to intersect the line defined by d and α in a tight cluster around the true velocity. Any constraint lines that are from different regions of different motion will intersect the line defined by d and α over a broad range of positions. The incorrect constraint lines generated along a motion boundary will not intersect the line defined by d and α at a consistent point.

The formula for the position of intersection along a constraint line can be easily derived by first rotating the coordinate system so that the constraint line defined by d and α is parallel to the vertical axis in velocity space. Let the angle between the two constraint lines be denoted by $\phi = \alpha' - \alpha$ as shown in Figure 3.4. The distance of the intersection along the constraint line may be computed by,

$$b = \frac{c}{\tan(\phi)}. \tag{3.25}$$

The distance c is related to d, d' and ϕ by,

$$(d + c) \cos(\phi) = d'. \tag{3.26}$$

Using Equation 3.25 to eliminate c from Equation 3.26 yields,

$$d\cos(\phi) + b\sin(\phi) = d'. \tag{3.27}$$

Solving for b and substituting using the definition for ϕ yields,

$$b = \frac{d'-d\cos(\phi)}{\sin(\phi)} = \frac{d'-d\cos(\alpha'-\alpha)}{\sin(\alpha'-\alpha)}. \tag{3.28}$$

This formula provides the position along a constraint line defined by d and α of the intersection of the constraint line with another constraint line defined by d' and α'. So given a set $\{ b_i \}$ of intersections of the constraint lines within a neighborhood with a constraint line at the center of the neighborhood, the set of intersections may be analyzed to determine the most consistent subset of intersections that cluster about the likely velocity. The cluster analysis criterion is motivated and explained by imagining a motion boundary passing almost vertically through a neighborhood just to the left of a center element. At any reasonable pixel resolution, the boundary will most likely pass smoothly through the neighborhood dividing the neighborhood almost in half. The region on one side corresponds to one surface in the scene and the other side including the center element corresponds to a different surface. In the extreme case where the boundary passes very close to the center of the neighborhood, almost half of the intersections of neighborhood constraint lines with the constraint line from the center of the neighborhood will be with constraints from the other side of the surface or the motion boundary itself. These intersections will most likely not be close to the velocity of the surface from which the center constraint line was obtained. Almost half of the intersections may be useless in the most extreme case; but at least half of the intersections will correspond to the same

region of motion. Since rejecting correct intersections is less harmful than accepting incorrect intersections, a conservative algorithm can be taken: the algorithm looks for the tightest cluster that contains roughly half of the intersections. The cluster analysis is done in one-dimension which makes it easy. The algorithm sorts the set of n intersections $\{b_i\}$ and then looks for the tightest interval that contains half of the intersections by examining successive pair of intersections that are [n/2] intersections apart. The algorithm chooses the pair of intersections that are closest together as the estimate of the majority cluster of intersections. Then the center position b' of the tightest cluster along the constraint line is the midpoint of the smallest interval that contains roughly half of the intersections. So the estimated speed of motion is given by

$$\bar{v} = \sqrt{d^2 + \bar{b}^2}, \tag{3.29}$$

and the estimated direction of motion is given by,

$$\bar{\beta} = \alpha + \tan^{-1}(\bar{b} / d). \tag{3.30}$$

The estimated speed and direction of motion (v, β) computed with constraint line clustering avoids the error problem in motion boundaries and gives another solution to the image flow constraint equation. Also the constraint line clustering algorithm is very robust and works even if the basic assumption that at least half of the intersections must be from the same region of motion is violated. For example if the center constraint is from just inside a right angle corner so that only 1/4 of the intersections are from the same region of motion, the algorithm still works because the intersections from constraints outside the corner are almost uniformly distributed along the constraint line

and do not bias the estimate of the tightest cluster.

The constraint line intersections are not always numerically well conditioned. When two constraint lines are close in orientation there can be a large error in the intersection position. When the constraint lines have the same orientation and different displacements, the lines will not intersect. In spite of these objections, the constraint line clustering algorithm is robust by the fact that only the tightest group of points are retained to compute the motion estimate. Nearly half of the intersection positions will be discarded. Any grossly incorrect intersections will most probably be discarded since they lie outside the tightest cluster.

The algorithm is also very insensitive to the neighborhood size. Good results may be obtained even for the minimum neighborhood size of 3 by 3. Larger neighborhood sizes produce only slightly better motion estimates. There are two reasons for this. The first reason is that the accuracy of the motion estimate is limited by the accuracy of the constraint line along which the motion estimate is obtained. The second reason is that the number of points in the neighborhood may have little impact on the quality of the motion estimate; it is the spatial extent of the neighborhood relative to the gradient variation that determines the quality of the motion estimate. In order to get good intersection measures, motion constraints must be used from beyond the region of similar gradient. The motion estimate can be greatly improved by increasing the spatial coverage of the neighborhood. One apparent weakness with the constraint line clustering is that it depends on the accuracy of the constraint line at the center of the neighborhood. This is not a problem with the constraint line clustering algorithm itself but is an instance of a

more fundamental problem with assigning a motion estimate to a point in an image. If the constraint equation is grossly incorrect, then there is a fundamental ambiguity in assigning a motion estimate to the corresponding image location.

The information in image flow constraint equations could be combined by least squares methods. The image flow constraint equation is one constraint with two unknowns. The equations in a neighborhood could be solved by least squares methods for an initial motion estimate.

IV. ONE-DIMENSIONAL FOURIER TRANSFORM FOR TRACKING MOVING OBJECTS

A. INTRODUCTION

The method presented in this chapter brings a solution to the problem of determining motion estimates via a One-Dimensional Fourier transform formulation. The technique identifies objects uniquely by their motion using an algorithm which is based on the application of the one-dimensional Fourier transform [Ref. 8:pp. 383-388].

This method not only separates the time-varying and time invariant parts of an image but perform these operations with the least amount of computation. The method presented here associates sinusoids with the time-varying parts of the image, such that the faster moving objects are associated with high frequency sinusoids and the time-invariant parts are associated with constant levels.

B. THE COSINE-AREA TRANSFORM

Let a one-dimensional discrete sequence be described by $f(x,t)$, $t=0,1,\ldots,T-1$ of T frames of size M. Define a general transformation as,

$$g(t) = \sum_{x=0}^{M-1} f(x,t)\ w(x|k) \qquad t=0,1,\ldots,T-1, \tag{4.1}$$

where $w(x \mid k)$is a weighing function of order k. A properly chosen weighing function will allow us to use this weighted area, $g(t)$, to find the velocity of a moving object while

discriminating against a stationary background.

If $w(x \mid k) = \cos(2\pi kx/M)$ is chosen as a weighting function, Equation 4.1 becomes,

$$g(t) = \sum_{x=0}^{M-1} f(x,t) \cos(2\pi kx/M) \qquad t=0,1,...,T-1. \tag{4.2}$$

So if the signal $f(x,t)$ is multiplied point for point with the amplitude of a sinusoid and the result is summed (integrated) over all the spatial variables $(0,M)$, the result is a single value $g(t_i)$ at each point t_i. When the signal is constant with respect to time, $g(t)$ will have a constant value, i.e. $g(t) = A$. When the time sequence contains a moving object, $g(t)$ will trace out a sinusoid [Ref. 9:pp. 281-283].

Suppose that for frame one (t=0) the sequence yields a one-dimensional array with M entries that are zero except at one location x_m (where the object is). So the weighted sum gives $g(0)=\cos(2\pi kx_m/M)$. The movement of the object by one location in the next frame (t=1) gives $g(1)=\cos(2\pi k(x_m+1)/M)$. If the object continues to move one location per frame, $g(t)$ yields a sinusoid with frequency of k/M. If the object were moving v_x locations between frames then the sinusoid would have a frequency v_xk/M. Since t varies between 0 and $T-1$ in integer increments, if we restrict k to have integer values then the discrete Fourier transform of the sinusoid would have 2 peaks, one located at frequency v_xk and the other at $T-v_xk$. This latter peak is due to the symmetry foldover of the Fourier transform. Thus a peak search in the Fourier spectrum would yield v_xk and division of this quantity by k yields v_x.

The factor k in the weighting function has several properties and limitations.

Fundamentally,

$$0 < |k| < M / 2, \tag{4.3}$$

which is basically determined by Nyquist sampling criteria to prevent aliasing and for convenience k is usually a positive integer. A zero k value allows no sinusoid to be created by a moving object and a k greater than $M/2$ produces an aliased spectral response. The actual value of k is chosen from the frame rate and the maximum expected object velocity,

$$k = f_{max} / v_{max}, \tag{4.4}$$

where f_{max} is the frequency limitation given by the available number of frames and the Nyquist criteria. The maximum and minimum velocities represent the dynamic range of the system, i.e.,

$$v_{max} / v_{min} = \#\ frames / 2. \tag{4.5}$$

The Fourier transform of $g(t)$ provides the basis for an efficient and effective method for estimating the angular frequencies of sinusoids. The Fourier transform of $g(t)$ is given by,

$$\begin{aligned} G(f) &= \sum_{t=0}^{T-1} g(t)\, e^{-j2\pi ft/T} \qquad f=0,1,\ldots,T-1 \\ &= \sum_{t=0}^{T-1} \sum_{x=0}^{M-1} f(x,t)\, \cos(2\pi kx/M)\, e^{-j2\pi ft/T}. \end{aligned} \tag{4.6}$$

The result is that the moving object generates a frequency proportional to its velocity.

All the stationary objects and the background will appear as a zero frequency component in this transform domain. The frequencies generated by the moving objects can be easily discerned by a simple peak detection algorithm.

This method is easily generalized for use on two-dimensional spatial images. Let the time sequence of images be represented by $f(x,y,t)$ with $0 \leq x \leq N$, $0 \leq y \leq N$, $0 \leq t \leq N$. The resulting velocity is a vector $v = [\, v_x \; v_y \,]^T$. The cosine area transform is defined by,

$$\begin{pmatrix} g_x(t) \\ g_y(t) \end{pmatrix} = \sum_{\substack{x=0 \\ y=0}}^{N-1} f(x,y,t) \cos\begin{pmatrix} 2\pi kx/N \\ 2\pi ky/N \end{pmatrix} e^{-j2\pi ft/T}, \tag{4.7}$$

and to extract velocity components, the following Fourier transformations are taken,

$$\begin{pmatrix} G_x(f) \\ G_y(f) \end{pmatrix} = \sum_{t=0}^{T-1} \begin{pmatrix} g_x(t) \\ g_y(t) \end{pmatrix} e^{-j2\pi ft/T}. \tag{4.8}$$

C. THE GENERALIZED AREA-TRANSFORM

We will generalize the cosine area transform in a way that will yield us the extraction of both velocity components simultaneously and identification of the moving objects in the original sequence. Two Fourier transformations are the foundation for this generalization. These transformations result from a replacement of the $\cos(2\pi kx/N)$ term with $\exp(-j2\pi kx/N)$. The first transform is a space to inverse-space transform whose values are subsequently summed over the second spatial variable; while the second is a time to frequency transform [Ref. 9:p. 284]. The transformations for a sequence of T

digital images of size $N \times N$, at any integer instant of time are given by,

$$F(k_x,t) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y,t)e^{\frac{-j2\pi k_x x}{N}} \qquad k_x=0,\pm1,... \tag{4.9}$$

$$F(k_y,t) = \sum_{y=0}^{N-1}\sum_{x=0}^{N-1} f(x,y,t)e^{\frac{-j2\pi k_y y}{N}} \qquad k_y=0,\pm1,... \tag{4.10}$$

$$G(k_x,f) = \sum_{t=0}^{T-1} F(k_x,t)\ e^{\frac{-j2\pi ft}{T}} \qquad f=0,1,...,T-1 \tag{4.11}$$

$$G(k_y,f) = \sum_{t=0}^{T-1} F(k_y,t)\ e^{\frac{-j2\pi ft}{T}} \qquad f=0,1,...,T-1 \tag{4.12}$$

It should be noted that separate transforms are used for each spatial dimension, and the first two transforms are summed over both spatial variables. This is done so that the velocity components can be resolved for each dimension. The velocity k_u product is found by detecting a peak in the spectrum of $G(k_u,f)$ for a fixed value of k_u. The velocity of an object in the dimension u is then found by dividing the frequency of the peak by k_u. Also it is of interest to note that a sequence of frames in which no motion takes place would yield identical exponential terms whose Fourier transform would consist of a single peak at a frequency of 0. Since the operations are linear, the general case involving one or more moving objects in an arbitrary static background would have a Fourier transform with a peak at 0 (corresponding to the static image components) and peaks at locations proportional to the velocities of the objects.

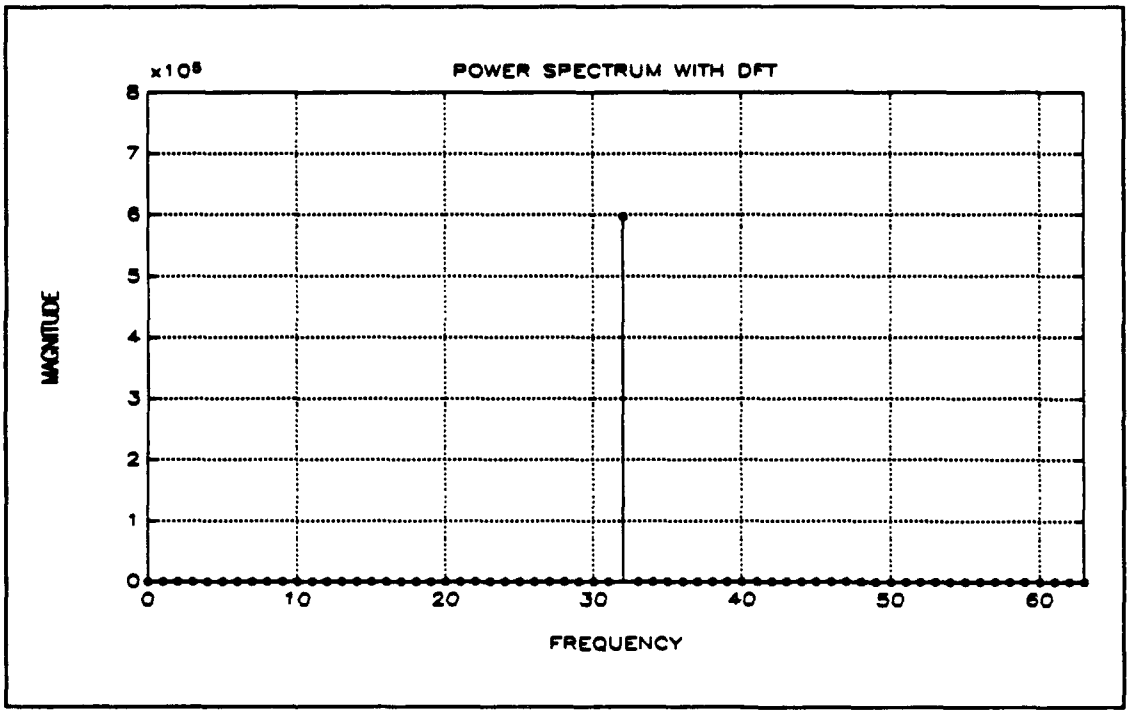


Figure 4.1 The ideal power spectrum ($G(k_u,f)$) with DFT.

The determination of a velocity component requires the use of phase information derived from the real and imaginary parts of $F(k_u,f)$. Since the speed of object was directly obtained from the spectrum of $G(k_u,f)$, the phase information in $F(k_u,t)$ is used to derive the sign of the velocity. Once a moving object is found, comparing the signs of the equations below determines the sign of the velocity;

$$S_1 = \left. \frac{d^2 Re\{F(k_u,t)\}}{dt^2} \right|_{t=t_1}, \tag{4.13}$$

$$S_2 = \left. \frac{d^2 Im\{F(k_u,t)\}}{dt^2} \right|_{t=t_1}. \tag{4.14}$$

Since $F(k_u,t)$ is a sinusoidal, S_1 and S_2 will have the same sign at an arbitrary point in

time t_1 if the velocity has a positive sign. The velocity will be negative if S_1 and S_2 have opposite signs. It should be noted that the ambiguity resulting when S_1 or S_2 equals zero is avoided by using the closest point in time $t = t_1 \pm \Delta t$.

The algorithm proposes that the 1-D FFT (DFT) spectrum of $F(k_x,t)$ provided by Equation 4.11, 4.12 will have a peak proportional to the velocity components of the moving object as shown in Figure 4.1. In reality the algorithm yields a wide spectrum. This spectrum has "smeared" frequency peaks instead of one frequency peak at the desired location as shown in Figure 4.2.

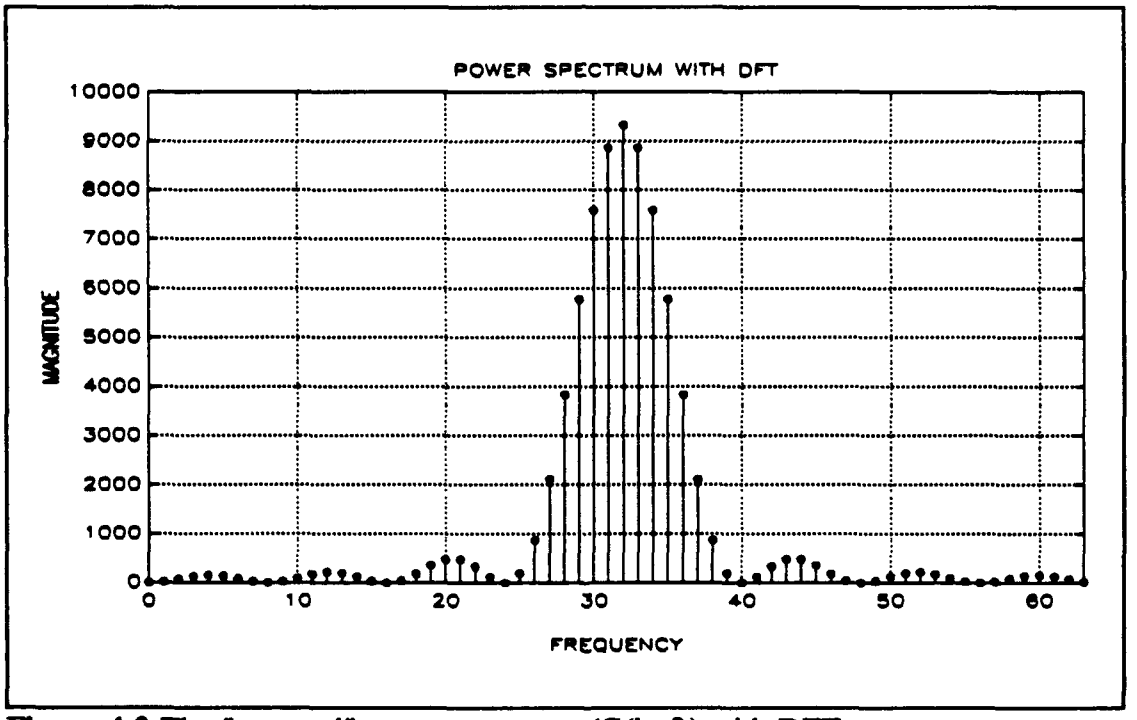


Figure 4.2 The "smeared" power spectrum ($G(k_x,f)$) with DFT.

There are mainly 2 reasons for this undesirable result.

(i) The undesirable signal components in the image frames (noise, occluding boundary effects, rotation of object, non-stationary background etc.). These "noise effects" can

cause spectral spreading.

(ii) The loss of frequency resolution due to limited number of frames.

The first condition has already been mentioned where the model did not include the non-stationary and non-uniform background. The effect of noise is always a problem for any algorithm.

The second condition is caused by the situations where the equation V_{max}/V_{min} = # *frames/2* is not satisfied due to the lack of available frames. Note that if V_{min} is taken to be 1 pixel per unit time, V_{max} is limited by the number of frames available. This limitation may be overcome by assigning the appropriate value to k_{μ}. For the case where image size is an integer multiple of the available number of frames, the maximum peak occurs at the frequencies integer divident of actual velocity. For other cases, the single frequency component at $k_{\mu}\mu$ will be "smeared" out to other frequencies. This smearing effect is defined as "leakage" [Ref. 10:pp. 459-474] and may cause errors in velocity estimates which also add to the errors due to the noise effects. Also as the leakage causes significant nonzero frequency components at the undesired frequencies (starting from adjacent frequencies), it also reduces the magnitude of the peak at the desired (or accurate) frequency.

A precise analysis is beyond the scope of the discussion here, but an intuitive idea of what is causing leakage may be obtained by looking at the time domain representation of sinusoids. The primary source of leakage seen in the FFT (DFT) of these sinusoids are the discontinuity introduced in the periodic extensions of complex sinusoid sequences by the missing partial cycles in available data. Missing partial cycles of the periodic

sinusoids occurs when the image size is not an integer multiple of the available number of frames. Note that the best result occur in cases where the number of frames is equal to the image size.

One solution to the problems introduced above is to use Spectral Estimation techniques as a last step to extract the frequency information from the available corrupted data. Besides the 1-D FFT (DFT), the Maximum Entropy Method can be used to improve estimates of peak frequencies. The Maximum Entropy Method (MEM) was originally devised by Burg [Ref. 11] to overcome the fundemental limitations of Fourier-based methods for estimating the power spectrum of complex sinusoids in the presence of additive noise.

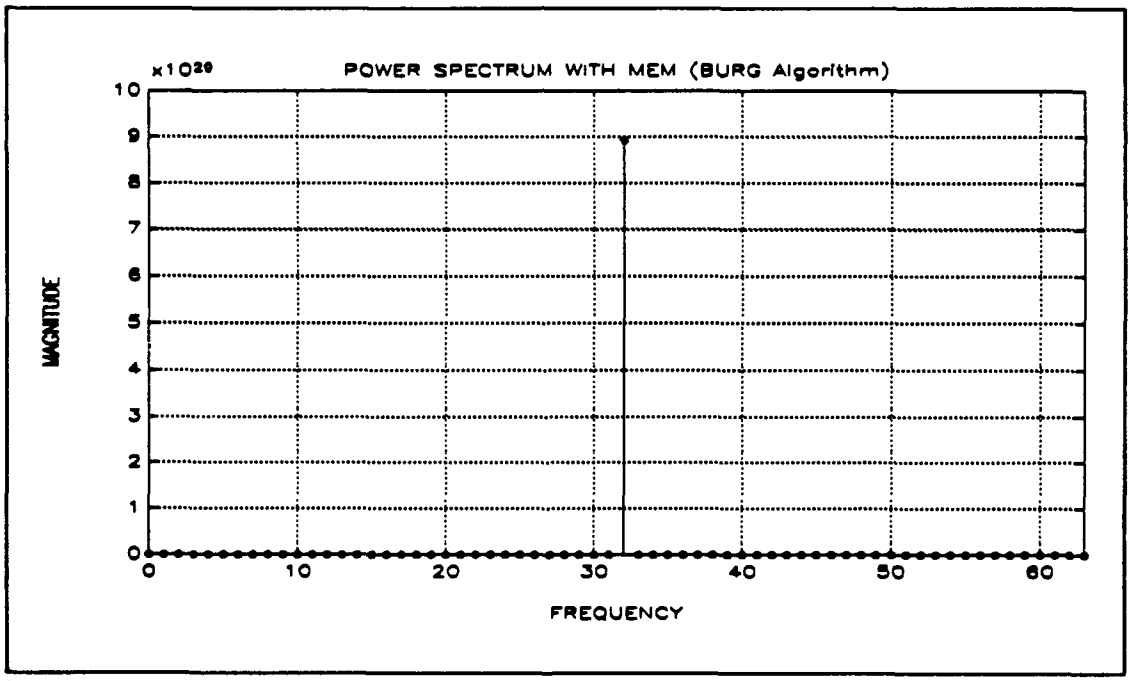


Figure 4.3 The improved power spectrum with the MEM method.

Other spectral estimation techniques such as the Multiple Signal Characterization (MUSIC) Algorithm, the Maximum Likelihood Method (MEM) and the Auto Regressive (AR) Signal Modelling were considered. The MEM was selected since it has significantly better resolution characteristics than the others when not much data is available [Ref. 12:pp. 385-392]. In particular, the MEM avoids the use of periodic extension of data or of the assumption that the data outside the available record length is zero.

V. THE 3-D FOURIER DOMAIN ANALYSIS OF IMAGE MOTION

A. THE SPATIOTEMPORAL FREQUENCY METHOD (STF)

This approach to image flow derivation takes the advantage of the special form of the Fourier transform of time varying images characterizing constant-velocity, rigid-body translation. The method is based on the spatiotemporal-frequency image representation and is called STF (Spatiotemporal-Frequency Approach) [Ref. 13,14]. A time-varying image function, moving in a spatial direction determined by the velocity vector $[u\ v]^T$ may be defined as,

$$\begin{aligned} f(x,y,t) &= f(x - ut, y - vt), \\ &= f(x,y) * \delta(x - ut, y - vt), \end{aligned} \tag{5.1}$$

where $f(x,y) = f(x,y,0)$, $\delta(x,y)$ is the Dirac delta function and "*" denotes convolution. The 2-D Fourier transform of Equation 5.1 incorporates a phase term of the form,

$$\begin{aligned} F_{2\text{-}D}\ \{g(x,y,t)\} &= e^{-j2\pi(w_x u + w_y v)t}\ F_{2\text{-}D}\{g(x,y,t)|_{t=0}\}, \\ &= e^{-j2\pi(w_x u + w_y v)t}\ F(w_x, w_y), \end{aligned} \tag{5.2}$$

where $F(w_x,w_y)$ is the fourier transform of $f(x,y)$. The existence of this phase component in the above equation enables the estimation of image velocity. This phase term is linearly dependent on t and the inner product of the 2-D Fourier indices $(w_x\ w_y)^T$ with the velocity vector $(u\ v)^T$. Determining the 3-D Fourier transform of Equation 5.1 only

requires an extension of the 2-D transform in Equation 5.2. Again assuming constant velocity, the 3-D Fourier transform can be denoted by,

$$\begin{aligned} F(w_x,w_y,w_t) &= F_{3\text{-}D}\{f(x,y,t)\}, \\ &= \int_{-\infty}^{\infty} F_{2\text{-}D}\{f(x,y,t)\}\ e^{-j2\pi w_t t}\ dt, \\ &= F(w_x,w_y)\int_{-\infty}^{\infty} e^{-j2\pi(w_x u+w_y v)t}\ e^{-j2\pi w_t t}\ dt, \\ &= F(w_x,w_y)\ \delta(w_x u+w_y v+w_t). \end{aligned} \tag{5.3}$$

As shown in Appendix B, this gives a very interesting result that a uniformly translating image with velocity $[u_o,v_o]^T$ has a Fourier spectrum that is zero elsewhere in the 3-D spatiotemporal space (w_x,w_y,w_t) except on the single plane defined by,

$$\{\ (w_x,w_y,w_t) : u_o w_x + v_o w_y + w_t = 0\ \}. \tag{5.4}$$

In general, for each velocity $[u_o,v_o]^T$ Equation 5.4 defines a unique planar locus of spatiotemporal frequencies which are consistent with that velocity as shown in Figure 5.1. Thus $F(w_x,w_y,w_t)$ may be visualized as a 3-D solid of constant intensities, where the slope of this solid surface is given by the values of u_o and v_o. This scheme for flow derivation however, does not provide the generation of an optical flow function; rather it yields a single velocity characterizing the entire image for all time. To make the computation of a time-varying and local image flow function possible, numerous regionally computed 3-D Fourier transforms could be employed instead of just one global transform. Then a regional velocity could be determined for each of the regional Fourier transforms in the manner described above.

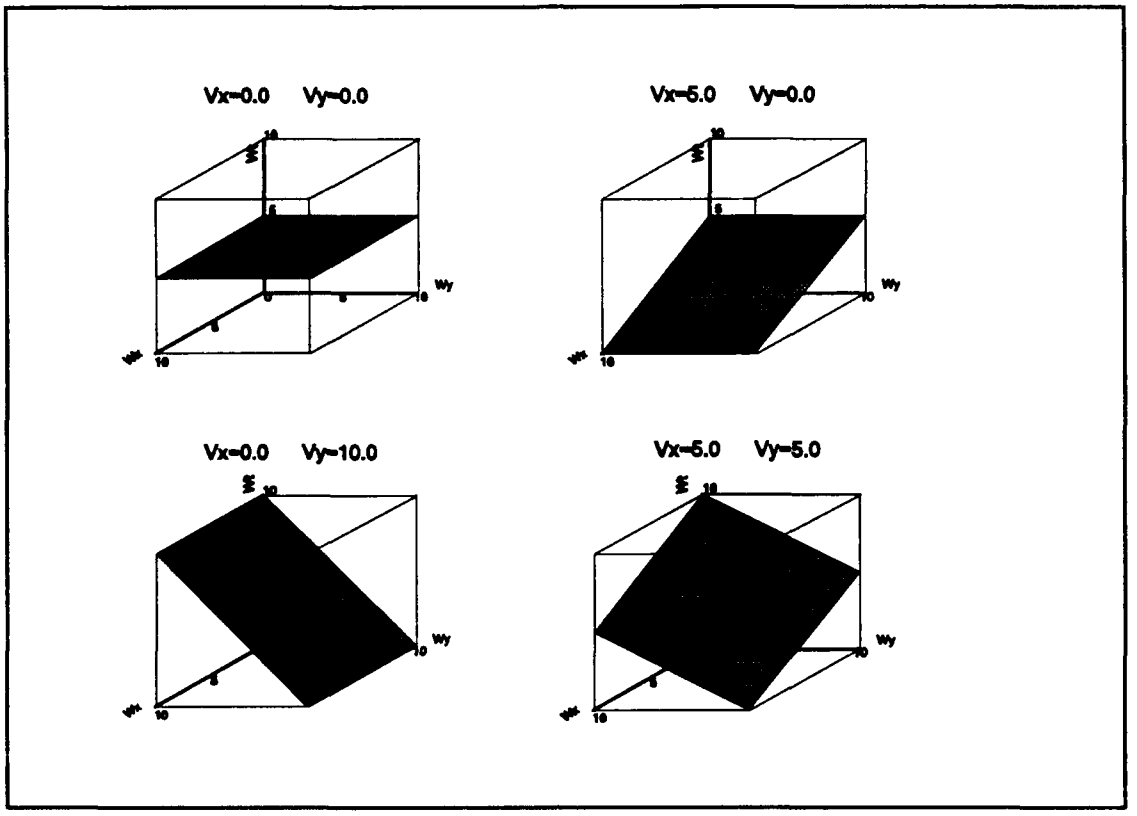


Figure 5.1 The velocity planes in the STF (spatiotemporal-frequency) Space.

The 3-D Fourier domain analysis of image motion is useful for several reasons, in particular:

1) It suggests a method for determining motion (velocity) parameters by examination of the nonzero region of $F(w_x, w_y, w_t)$.

2) It allows the design of a frequency domain filter for spatiotemporal interpolation of images [Ref. 4:pp. 61-66].

B. VELOCITY TUNED FILTERS

The form of velocity-tuned filters is motivated by the frequency-domain description of images which represent constant-velocity, rigid-body translation of objects [Ref. 15]. Although this is a very restrictive assumption, the results form a good foundation for considering more general types of motion. Motivated by the form of $F(w_x,w_y,w_t)$, which defines a plane in 3-dimensional frequency space, a velocity-tuned filter is defined as one which passes the Fourier transform of a constant-velocity function with unity gain and has zero gain over the rest of the frequency space. The filter will be supported on a unique planar surface defined by the velocity components of the image. However, such a filter turns out to have limited usefulness and is physically unrealizable. A better and more general approach is to use a finite bandwidth filter, one which has approximately unity gain in a solid 3-dimensional region surrounding the planar surface of $F(w_x,w_y,w_t)$ and goes to zero outside that region. So, we may consider the support of the 3-dimensional filter to be bounded by two planes which are parallel to the central plane and separated in the w_t direction by a temporal bandwidth of $F_t(w_t)$. The velocity-tuned filter will also pass constant-velocity functions whose velocity is different than the velocity to which the filter is tuned; this requires only that the filter transfer function $H(w_x,w_y,w_t)$ is approximately constant over the support of the input. The range of velocity over which this is true gets smaller as the temporal bandwidth of $F_t(w_t)$ decreases and as the spatial bandwidth of the input increases. Specifically the velocity range can be defined as,

$$| u^f - u^i | < \frac{B_t}{W_x} \quad \text{and} \quad | v^f - v^i | < \frac{B_t}{W_y}. \tag{5.5}$$

Here u^f, v^f are the velocity values to which the filter is tuned and u^i, v^i are the velocity values of the input [Ref. 15:p. 64], B_t is the temporal bandwidth of $F_t(w_t)$, and W_x, W_y are the spatial bandwiths of the original function. The quantities B_t/W_x and B_t/W_y collectively represent the effective velocity bandwidth of the filter when applied to this input. When the conditions of Equation 5.5 are not met, the Fourier transform of the input intersects the planar boundaries of the bandpass filter, and the frequency components outside the passband are attenuated.

VI. SIMULATION RESULTS AND CONCLUSIONS

The algorithms developed in the previous chapters are simulated with artificial and real image frames where MATLAB is used as a tool for simulation. The artificial image frames were 32x32 pixels in size and contained a square to represent the moving object. Sixteen shades of gray levels are used to represent the pixel intensity values of the square object and the background, although the intensity values are normalized before applying each algorithm. The algorithms are tested under two different background conditions. For the first case the image frames are assumed to have uniform background with zero intensity and for the second case a random checkerboard background with half the intensity of the moving object is used. A random checkerboard background is used in order to avoid any regular or periodic pattern in the background. The artificial images used in the simulations are shown in Figures 6.1-6.4.

Throughout the simulations, an artificial disturbance (noise) is added to the image frames (pixel values). The **Relative Error** in the estimation of velocity components is used as a comparison criteria for different **SNR** and different algorithms. The SNR (in decibels, dB) for an NxN image is defined as,

$$SNR = 10\,Log_{10}\left(\frac{\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)^2}{\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} n(x,y)^2}\right) \qquad (6.1)$$

where $f(x,y)$ is the desired component (actual pixel value), and $n(x,y)$ is the undesired component (noise). The final pixel values of the image frames are determined by the sum of these two terms. The **Relative Error** is defined as,

$$Relative\ Error = \frac{\|\hat{u}-u\|_2}{\|u\|_2} \tag{6.2}$$

where u is the actual value of velocity component, $\hat{u}$ is the estimated velocity component and $\|\bullet\|_2$ defines the Frobenius norm of a vector or matrix. Each algorithm is tested 4000-8000 times with very slowly changing SNR's. The resulting values of these tests are averaged over larger changes of SNR's. In each test, the algorithms are iterated 32 times. Since the image size was 32, the moving object is simulated such that as it disappeared from one end of the image frame it is allowed to reappear at the other end.

These algorithms are also tested with a real image sequence containing a moving car. This real image was 128x128 pixels in size and contained 256 shades of gray to represent the pixel values. These tests are done just to see if the algorithms would work with real images. Also since the EKF and the 3-D FFT algorithm (with Velocity Tuned Filter) promises restoration of the image and increase in SNR's, these two algorithms are tested with the real image sequences.

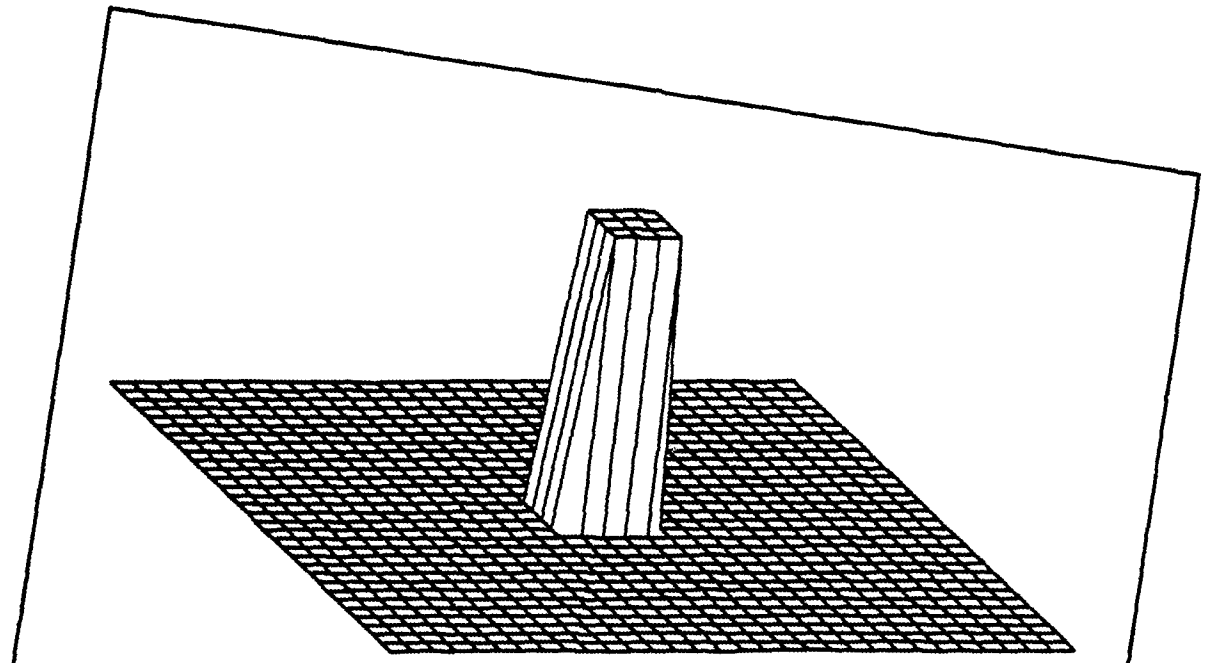


Figure 6.1 The test image frame containing a square object without noise.

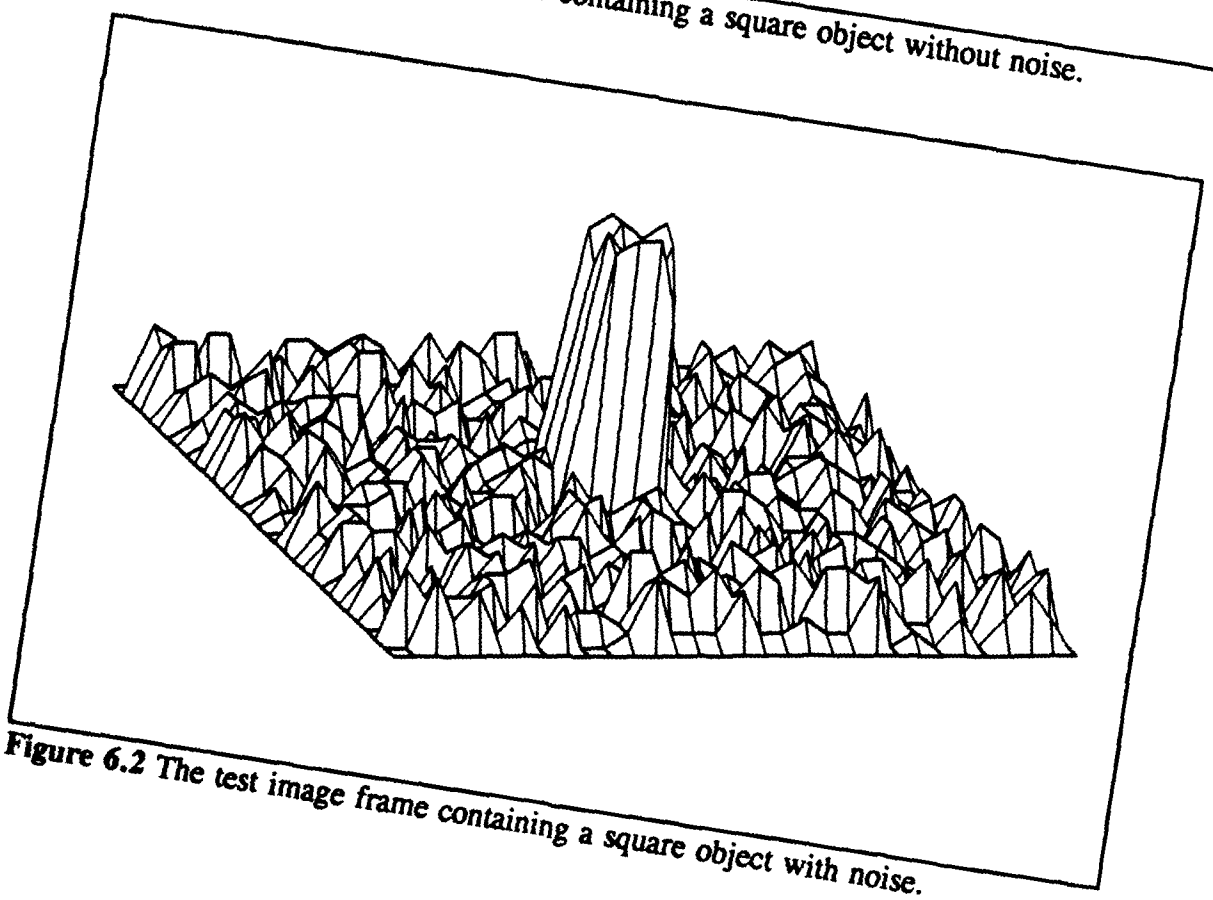


Figure 6.2 The test image frame containing a square object with noise.

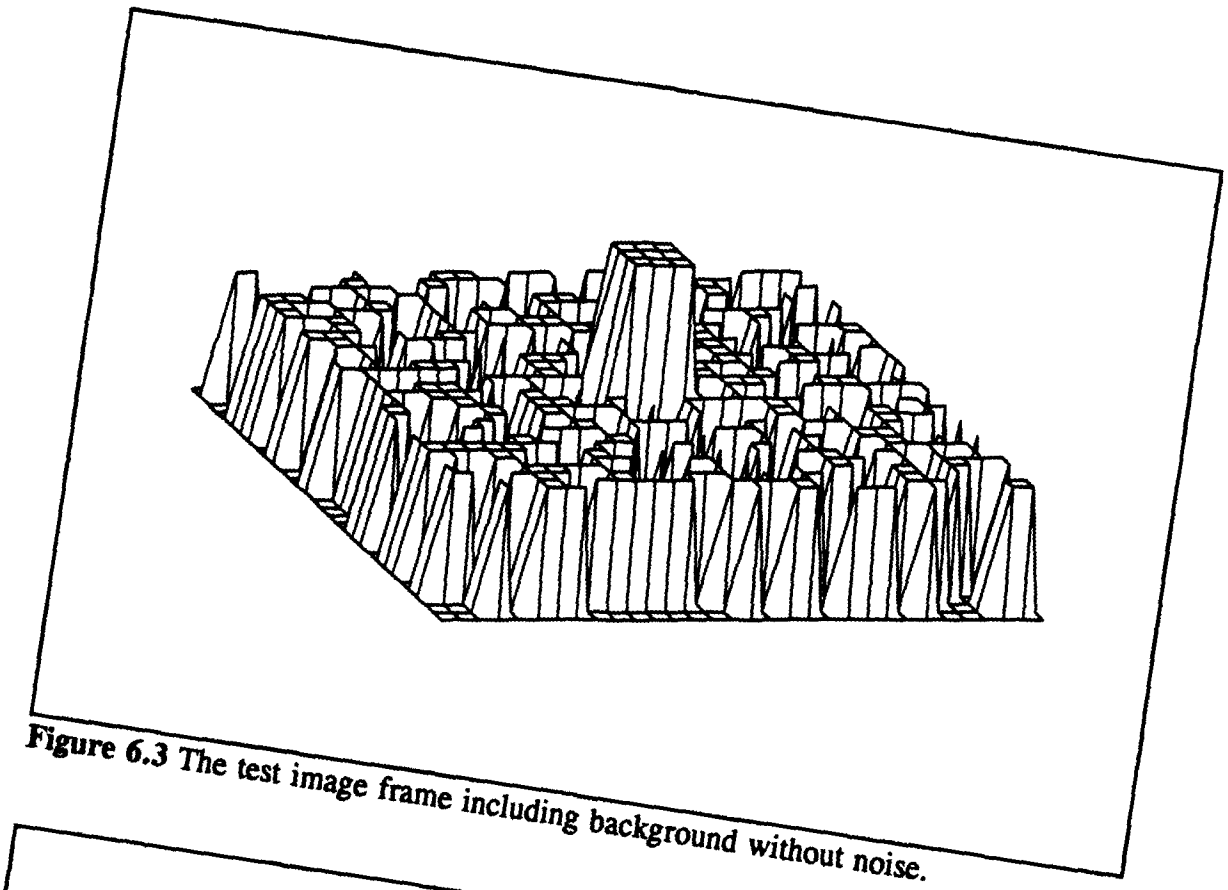


Figure 6.3 The test image frame including background without noise.

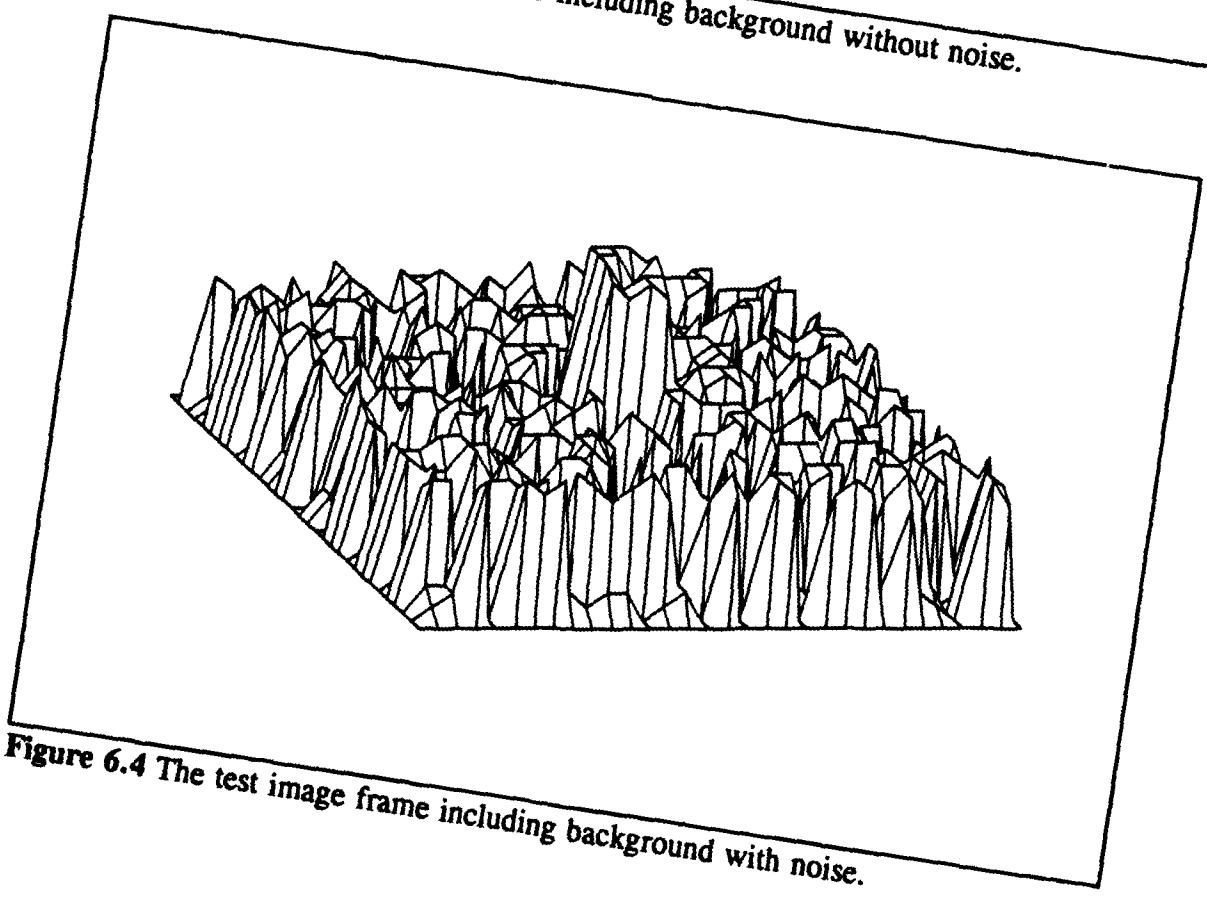


Figure 6.4 The test image frame including background with noise.

A. THE EXTENDED KALMAN FILTER

The extended Kalman filter is implemented with a 3rd order filter for images with zero (or uniform) background images and with a 5th order filter for images with nonzero background as was discussed in Chapter II.

Two sample test results of the EKF algorithm are shown in Figures 6.5-6.6. The EKF gives good results even under fairly low SNR conditions. In general at high SNR situations the EKF converges to the actual velocity values after 3-4 iterations and as the SNR decreases the iterations needed increases up to 25-30 iterations. At very low SNR conditions (-30,-40 dB), the EKF estimate of the velocity components is not guaranteed and depends on the random disturbance. Figure 6.7 shows the *Average Relative Error* (computed by averaging the relative error over 32 iterations) and the *Final Relative Error* (the relative error in the final velocity estimate) versus the *SNR* of the input image frames. Figure 6.8 shows the *Relative Error* versus *Iterations* for some selected values of *SNR*. Figures 6.9-6.10 shows similar results where the image frames included the background (5th order EKF). As seen from these two sets of results the background effected the performance of the algorithm. There are mainly two reasons for that. First, the occluding boundaries create a noise term which is not included in the *SNR* term. Second, the background which has much more signal power than the moving object, makes it harder to detect the phase changes due to the moving object. Similar effects are seen in other algorithms that use Fourier phase changes as a basis for velocity estimation.

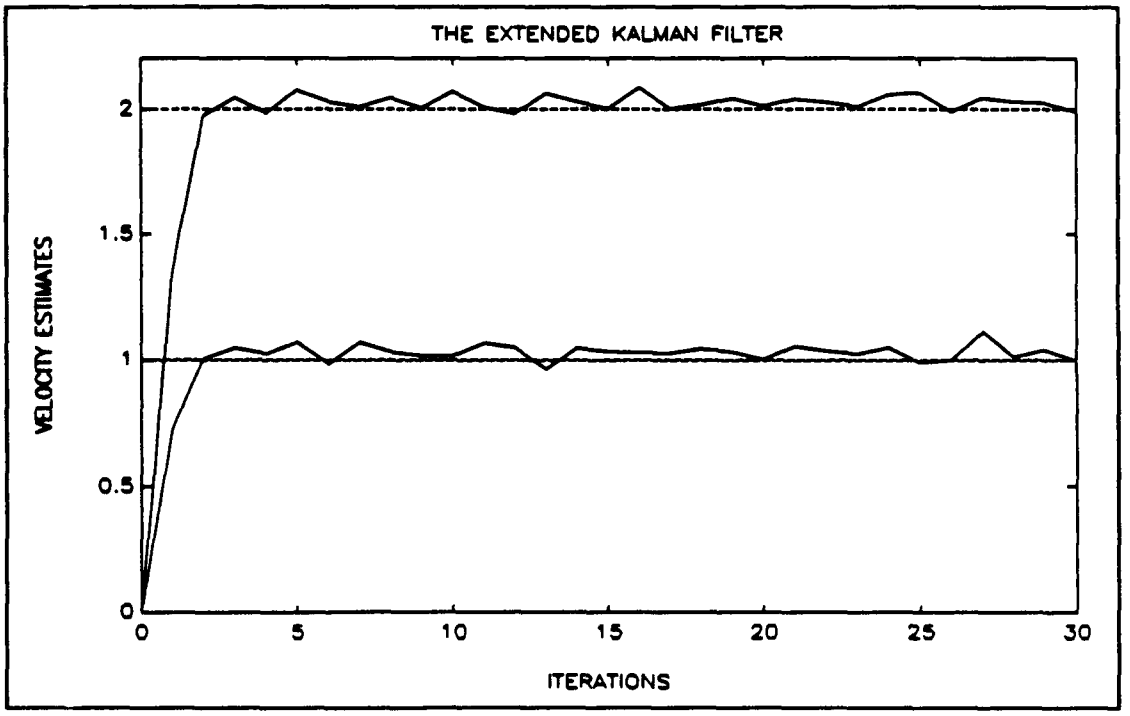


Figure 6.5 The EKF velocity estimates for a high SNR case.

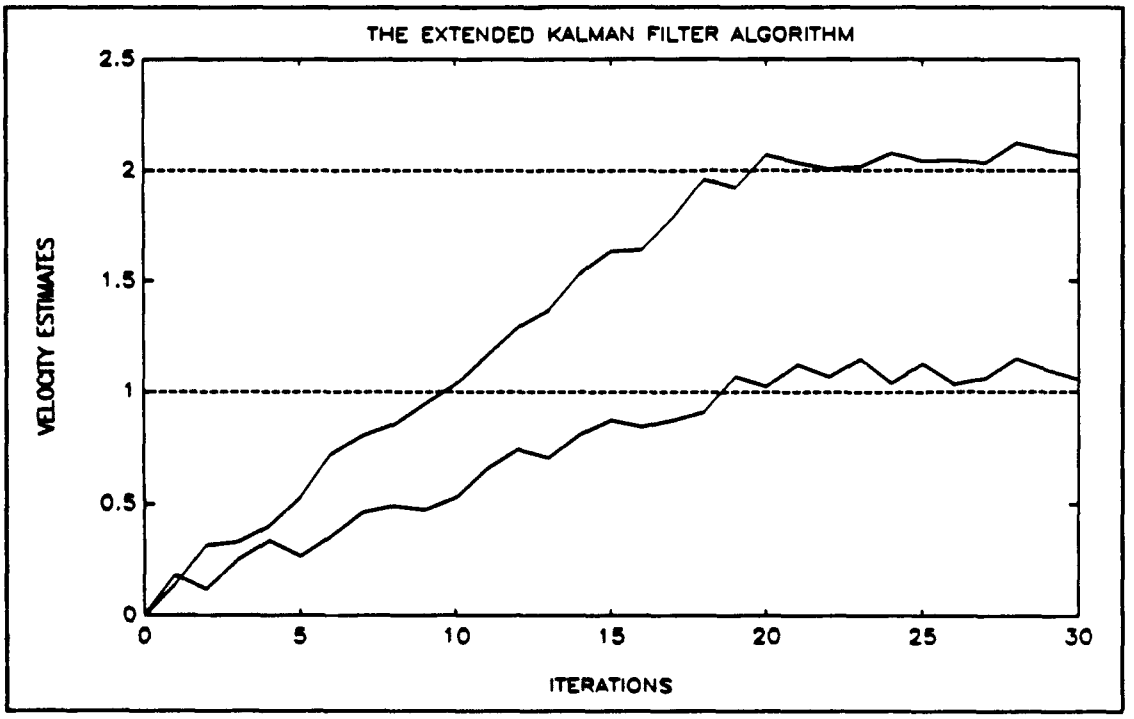


Figure 6.6 The EKF velocity estimates for a low SNR case.

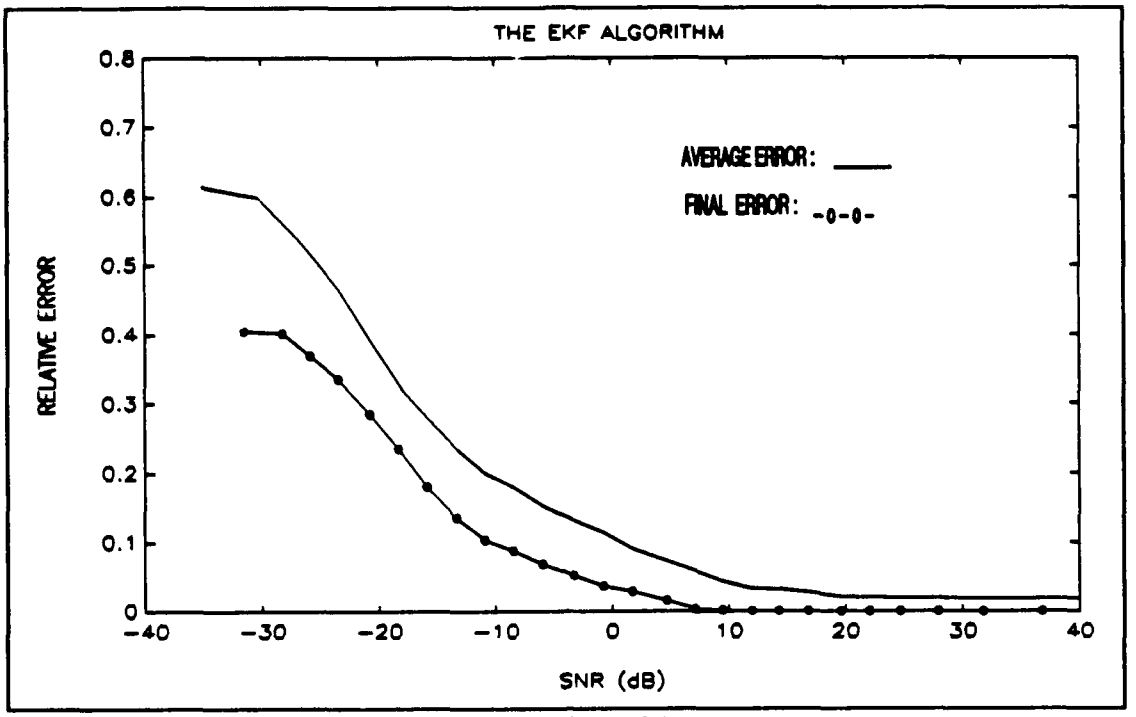


Figure 6.7 The Relative Error vs. SNR with a 3rd order EKF.

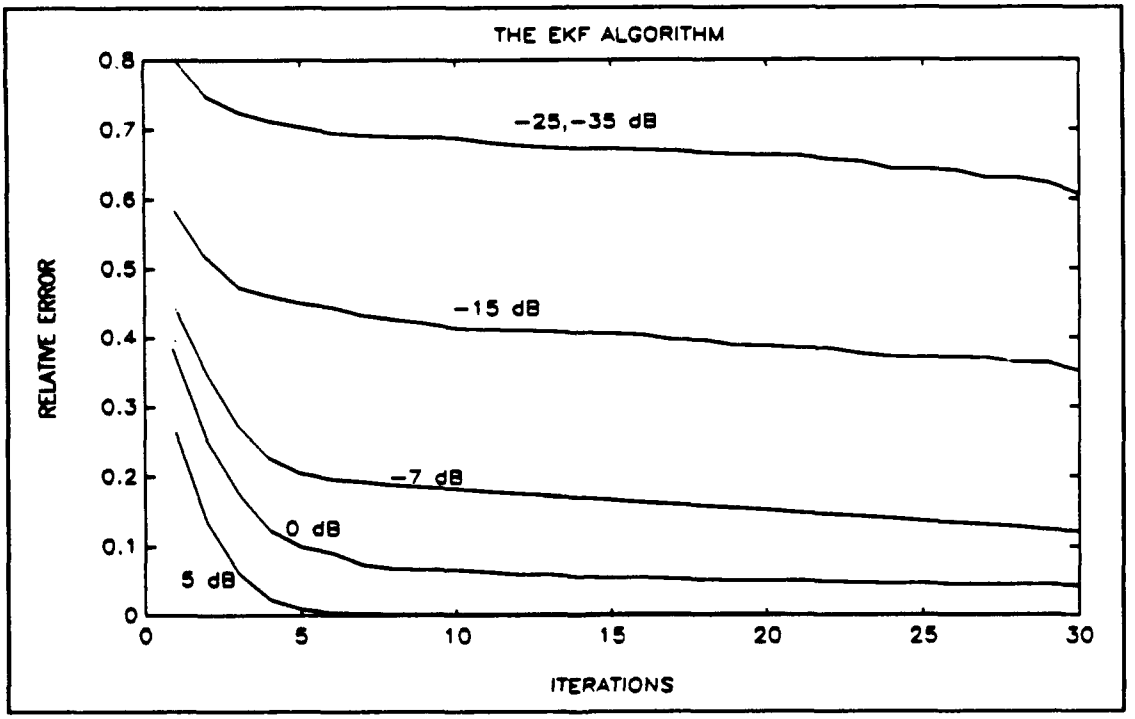


Figure 6.8 The Relative Error vs. Iterations for a 3rd order EKF.

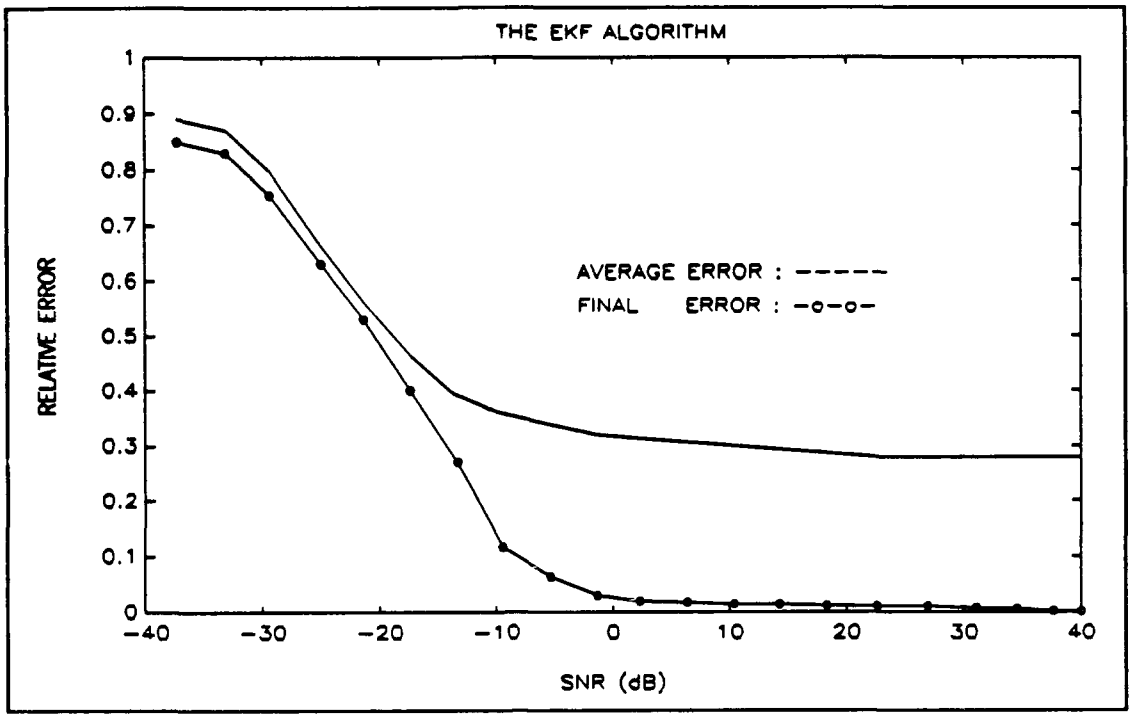


Figure 6.9 The Relative Error vs. SNR for a 5th order EKF.

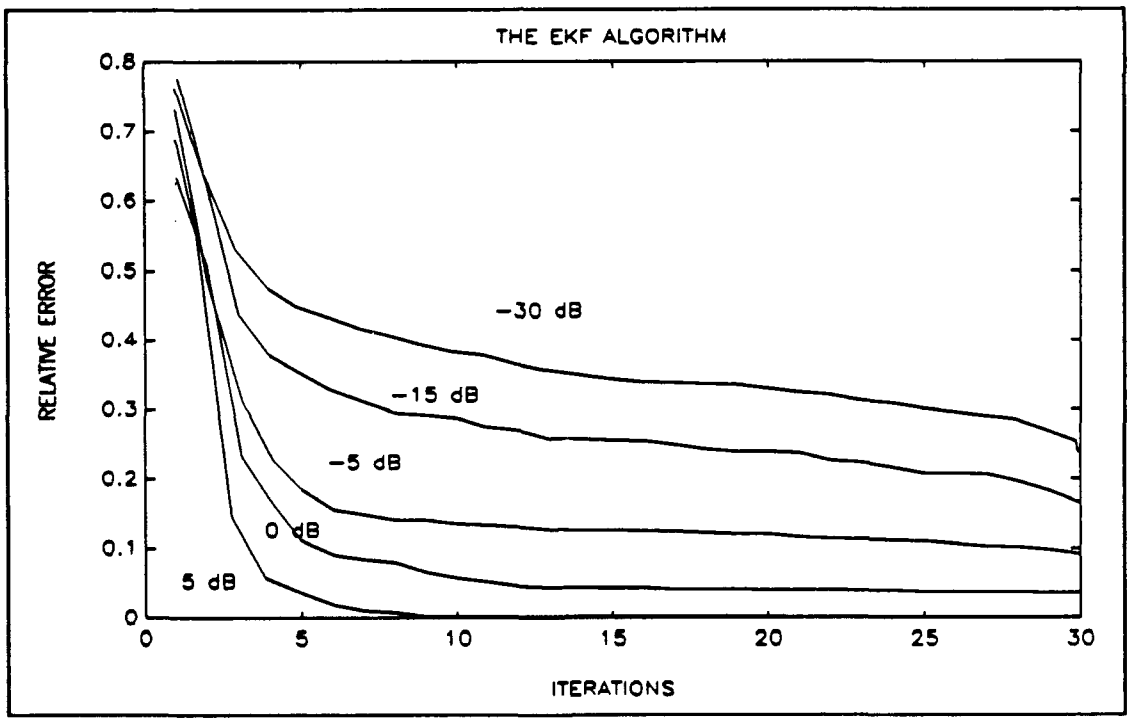


Figure 6.10 The Relative Error vs. Iterations for a 5th order EKF.

B. THE IMAGE FLOW CONSTRAINT EQUATION METHODS

The iterative solution given by Equations 3.16 - 3.17 and Constraint Line Clustering method are used for the computation of the Image Flow Constraint equation in the simulations. The iterative solution method is implemented in two different ways. For the first case only two frames are used and the computations are iterated for 1, 4, 16, and 32 times. The estimated velocity fields are shown as arrow diagrams in Figure 6.11. Initially, the velocity estimates are assumed to be zero. Consequently, the first iteration shows vectors where the temporal changes occur. Later the estimates approach the correct values in most parts of the image. A few changes occur after 16 iterations. In the next case, one iteration is used between two consecutive frames (per time step). The resulting velocity fields are shown in Figure 6.12 for 1, 4, 16, and 32 time steps. The estimates approached the correct values more rapidly and very few changes occur after 16 time steps. This leads to the final method where the velocity estimates are computed 4 times between two consecutive frames with 16 or 32 time steps. The velocity fields after 1, 4, 16, and 32 time steps, where 4 iterations per time step are used, is shown in Figure 6.13. The final estimates of the velocity components of the moving object are computed simply by averaging the velocity field in pixel locations that have significant temporal changes (i.e. where the moving object is). This created a thresholding problem to determine the region of motion especially for the high noise case. Since both the artificial image and the real image had higher intensity values at the object points than the background, the problem is solved rather easily. But as the noise

level is increased to higher values, the error in computing the regions of motion increased the error in the velocity estimates. Since the new estimates at a particular point do not directly depend on the previous estimates at the same point, increasing the time steps does not effect (or improve) the final velocity estimates. Although it improves the estimate of the velocity field.

Constraint Line Clustering methods do not use a previous velocity estimate at all. Consequently it is computed between two consecutive frames. Again the final velocity is estimated by averaging the velocity field in the region of motion. Figure 6.14 shows the velocity field computed by the Constraint Line Clustering algorithm.

Both algorithms show good performance when the noise levels are low even if the image has a finite number of discontinuities (or when the image is not smooth over all). However, the performance decreases quickly with increasing noise levels. Therefore, the image frames are smoothed with a Gaussian lowpass filter implemented in the frequency domain to improve the velocity field estimates. The Relative Error vs. SNR for these algorithms are shown in Figures 6.15-6.16 for image frames with 2 different choices of backgrounds respectively.

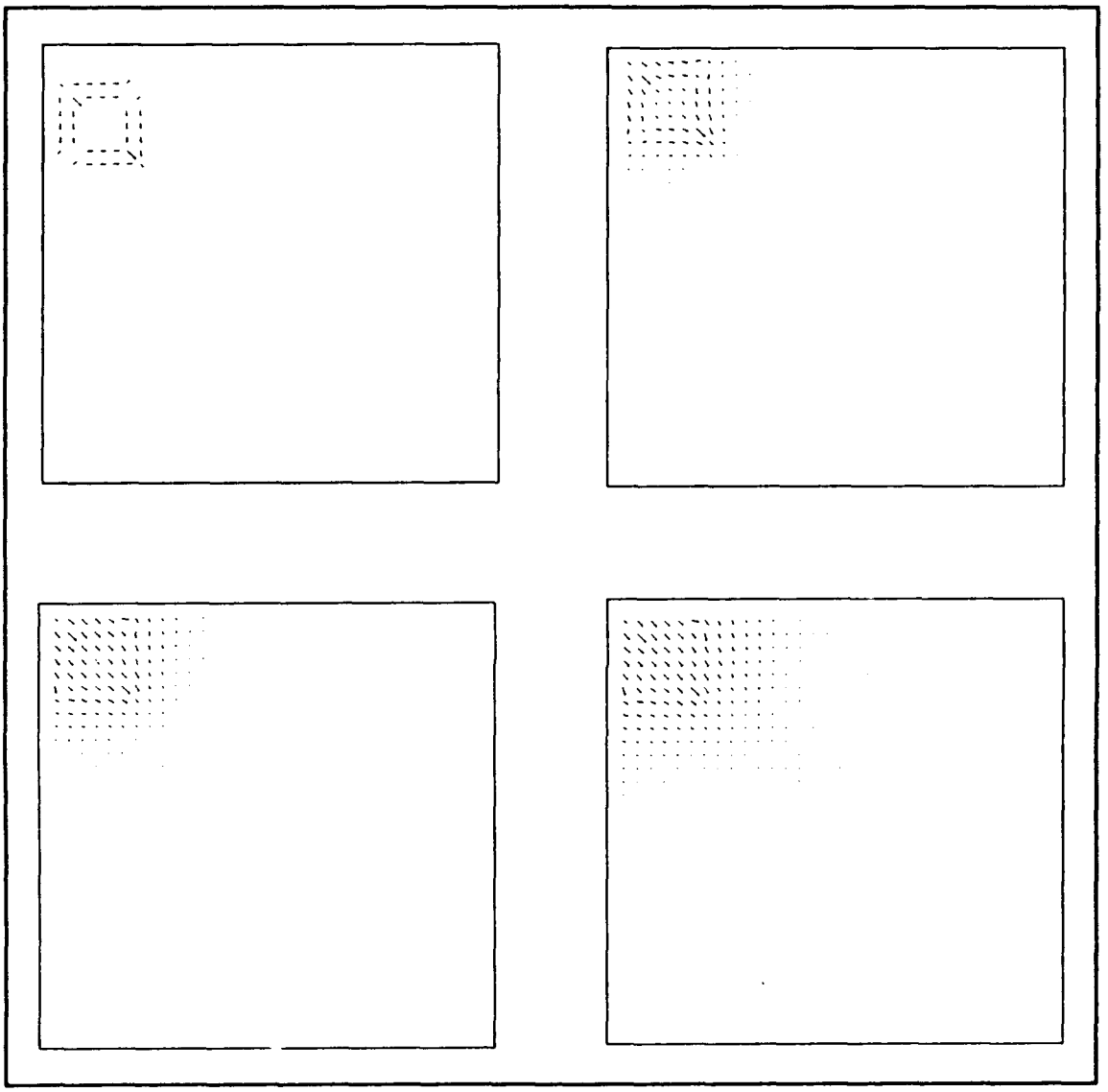


Figure 6.11 The velocity field after 1, 4, 16, and 32 iterations.

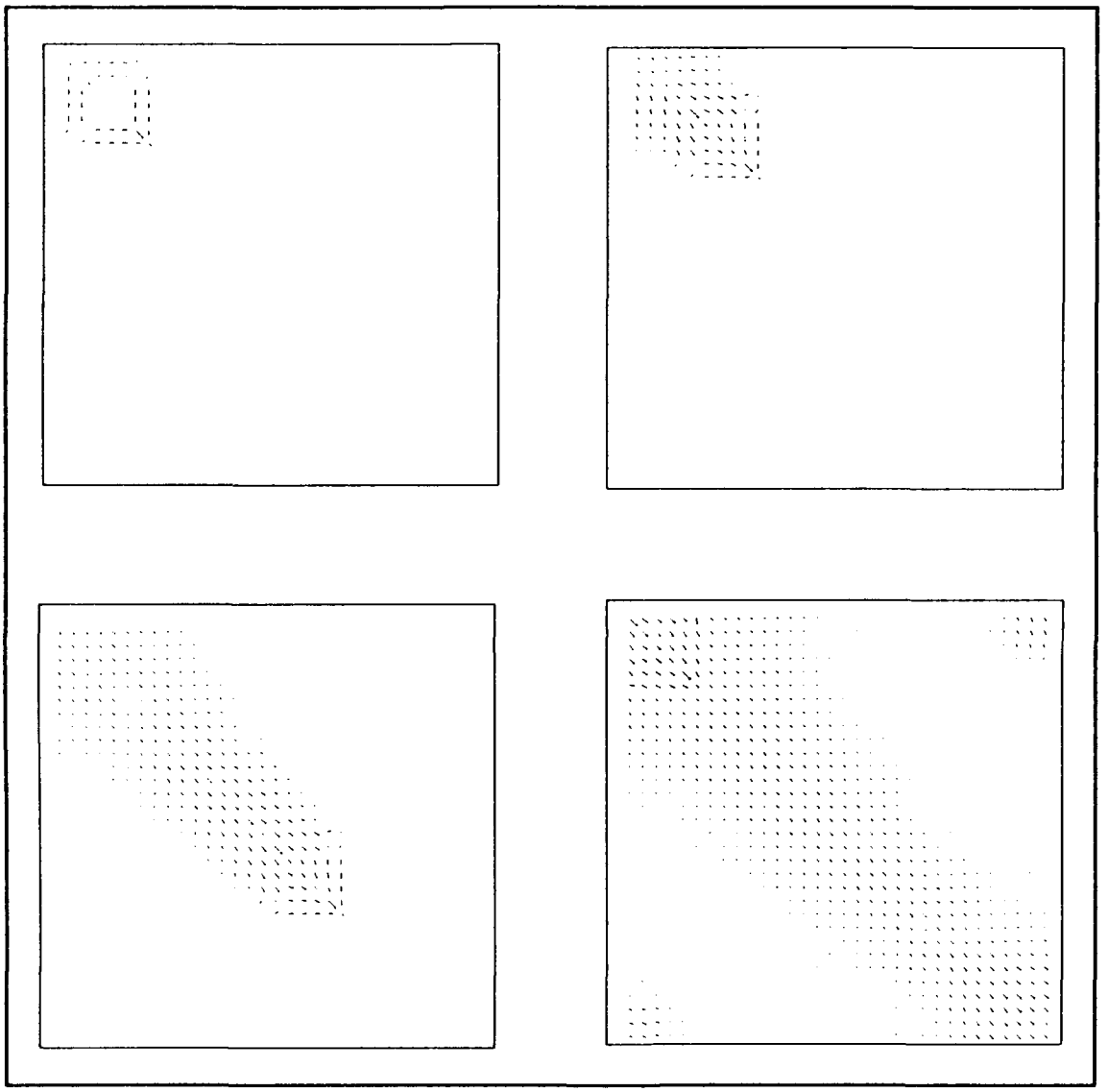


Figure 6.12 The velocity field after 1, 4, 16, and 32 time steps.

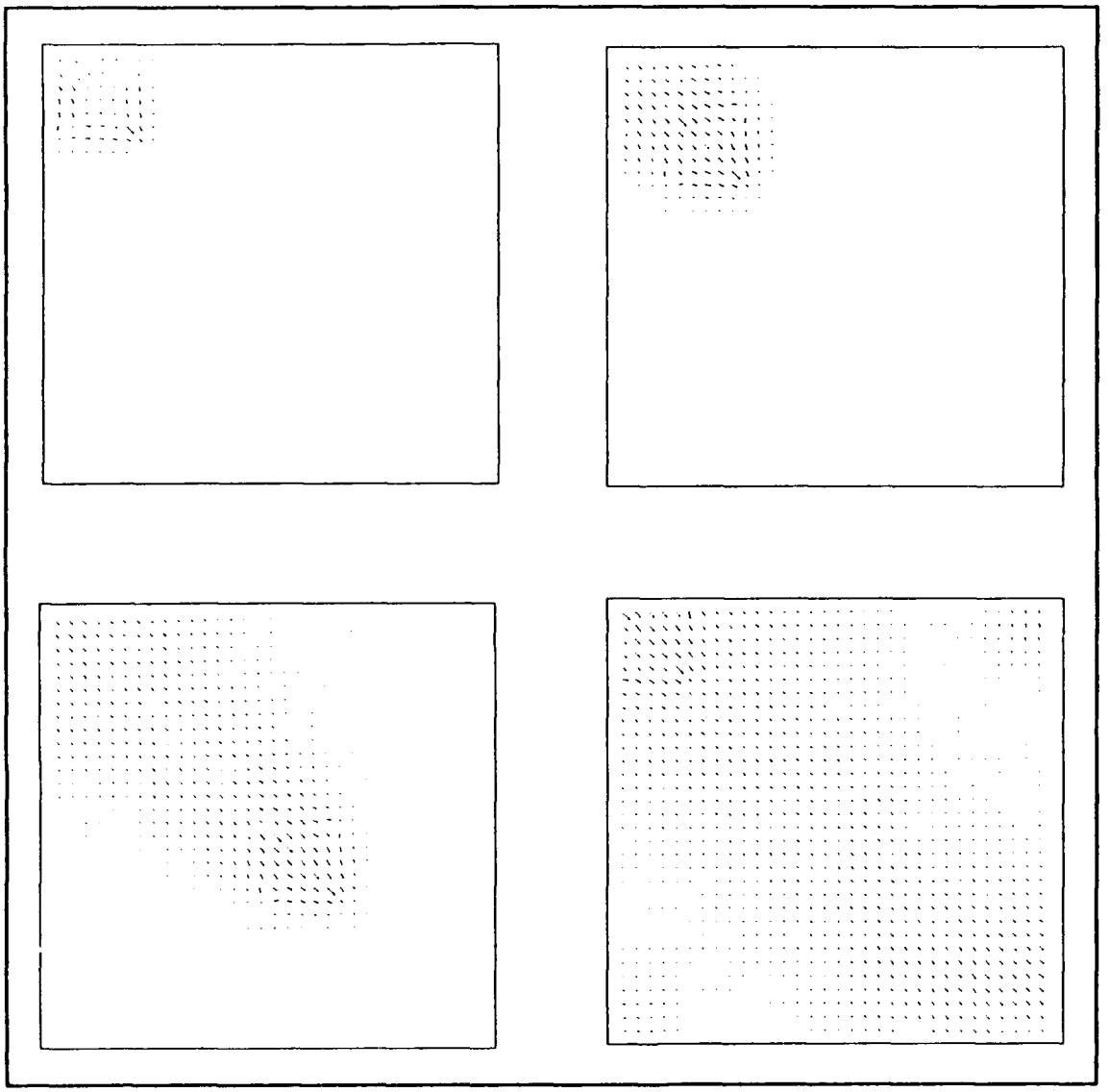


Figure 6.13 The velocity field after 1, 4, 16, and 32 time steps (4 iterations per time step).

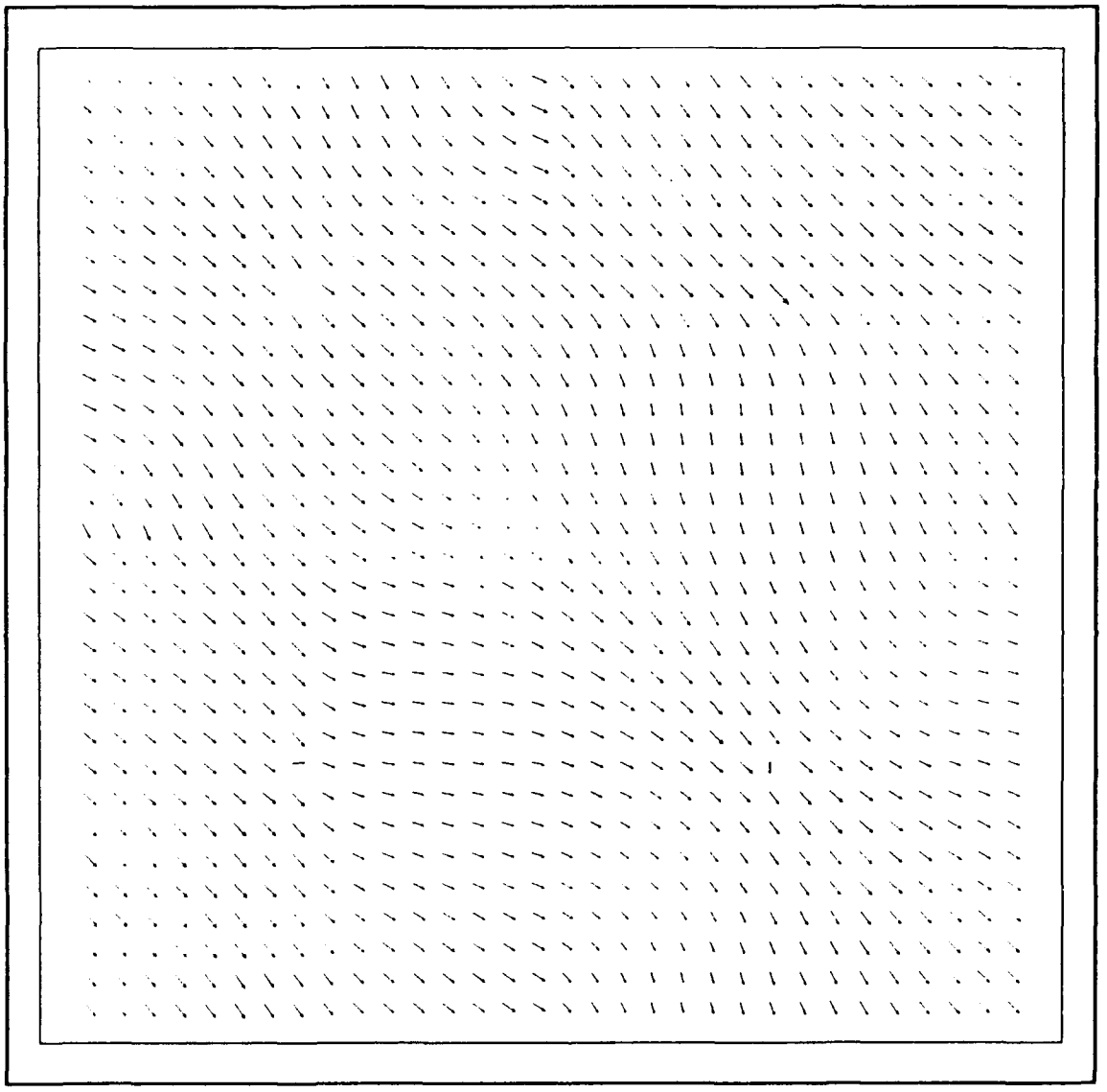


Figure 6.14 The velocity field with Constraint Line Clustering.

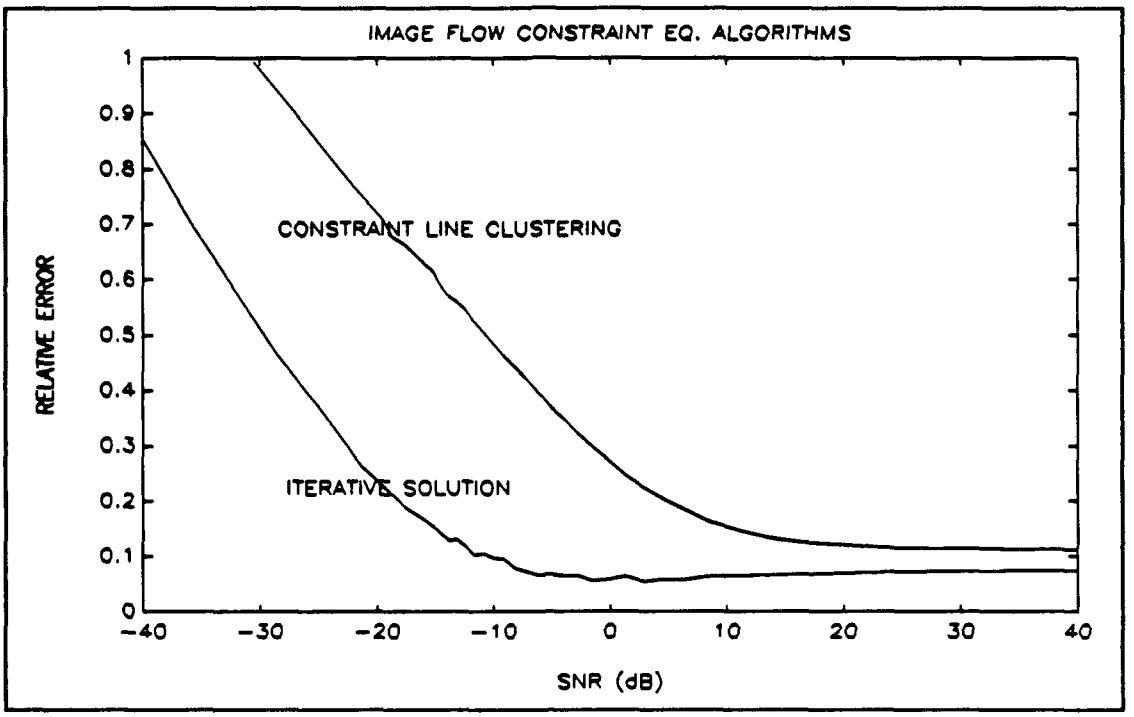


Figure 6.15 The Relative Error vs. SNR for Image Flow Constraint Eq. solutions.

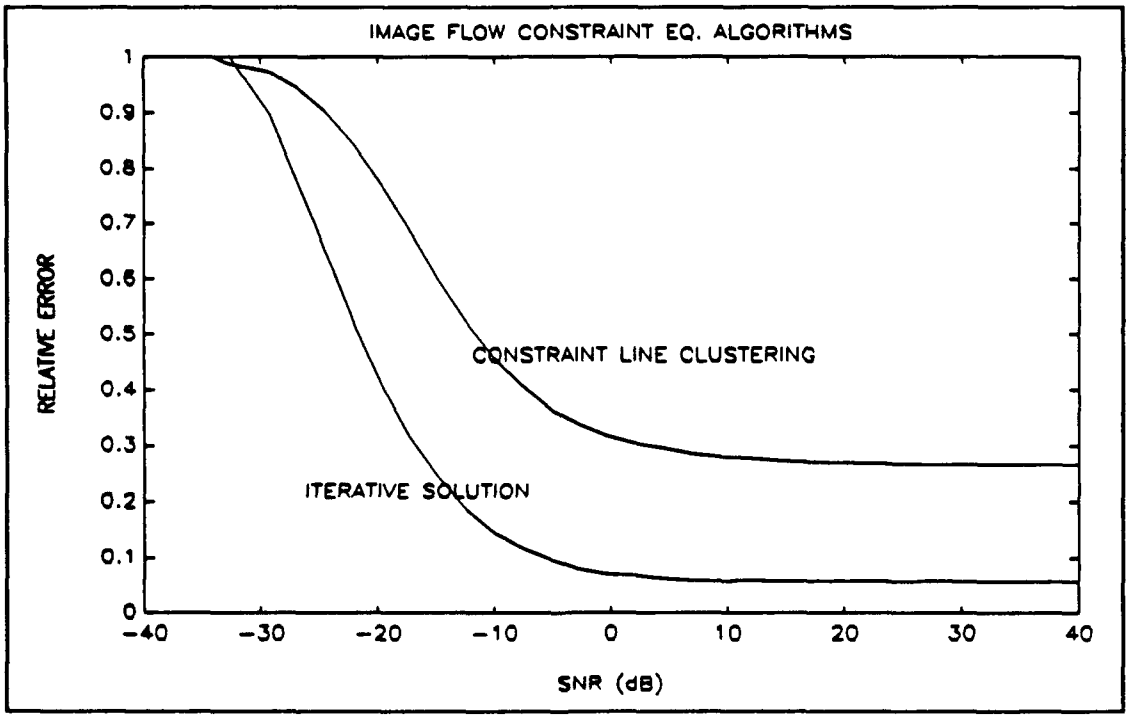


Figure 6.16 The Relative Error vs. SNR for Im. Flow Const. Eq. with background.

C. THE SPATIOTEMPORAL FREQUENCY (3-D FFT) ALGORITHM

This algorithm requires a large memory space since the previous image frames (or their 2-dimensional FFT's) need to be saved in order to perform the FFT in the third dimension. Also, the definition and detection of the plane in the 3-dimensional spatiotemporal frequency space is another problem to be solved. In the simulations the planes are defined by their slopes in the w_x, and w_y directions. These slopes are extracted by detecting the slope of the lines created by the intersection of the defined plane with the vertical planes normal to the w_x, and w_y axes. The slopes are determined as integer values which caused the final velocity estimates to be integer values as well. This is an acceptable situation since the actual velocity values are accepted as integer values too. The velocity estimates are affected by the number of frames available. Although it was not mentioned before, the effect of **smearing** (or leakage) is seen around the central plane because of missing image frames. Besides the expected nonzero central plane,the 3-D FFT creates other planes which are parallel to this central plane. These planes are also supported by the noise terms. One test of velocity estimation with the 3-D FFT algorithm is shown in Figure 6.17. Figures 6.18-6.19 show the Relative Error values vs. SNR and number of iterations for this algorithm.

The 3-D FFT algorithm can be used for images with nonzero background as well. In this case, the algorithm creates a second nonzero plane with the slope of zero due to the constant background. The actual plane can be determined by deleting the plane associated with the constant background first. The smearing effect is also seen around

the plane associated with the constant background. This makes the detection of the desired plane harder and creates errors in the velocity estimates. The simulation results of the Spatiotemporal Frequency algorithm with image sequences including background are shown in Figures 6.20-6.21.

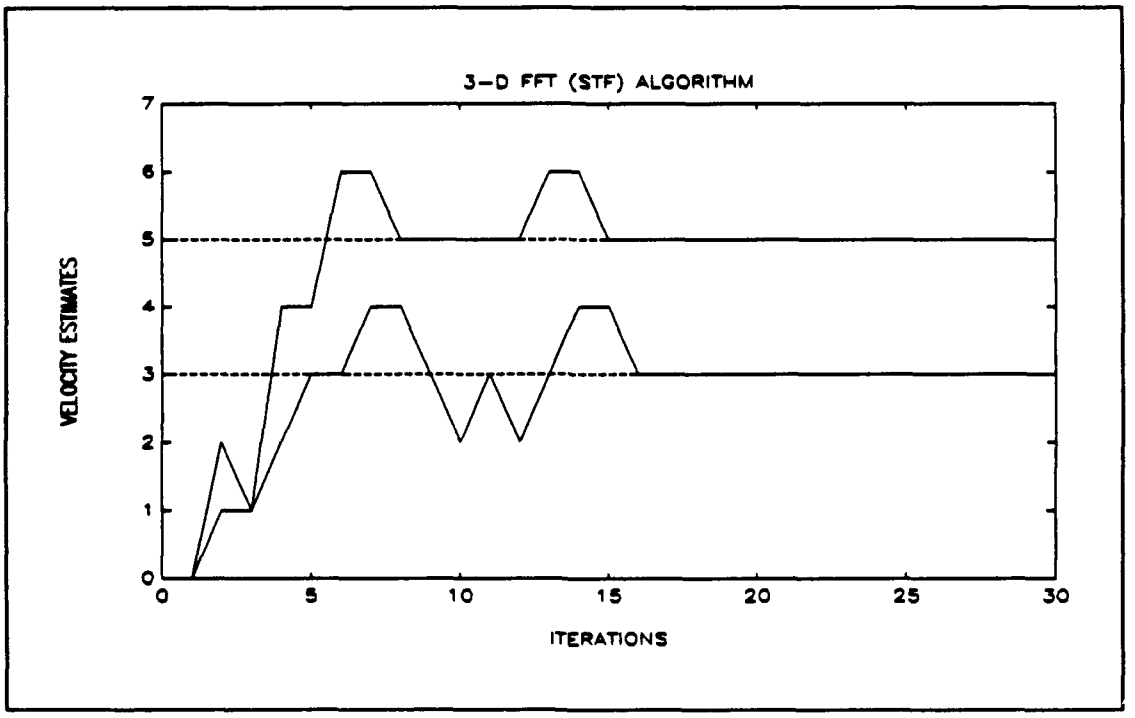


Figure 6.17 Velocity estimates with the Spatiotemporal Frequency algorithm.

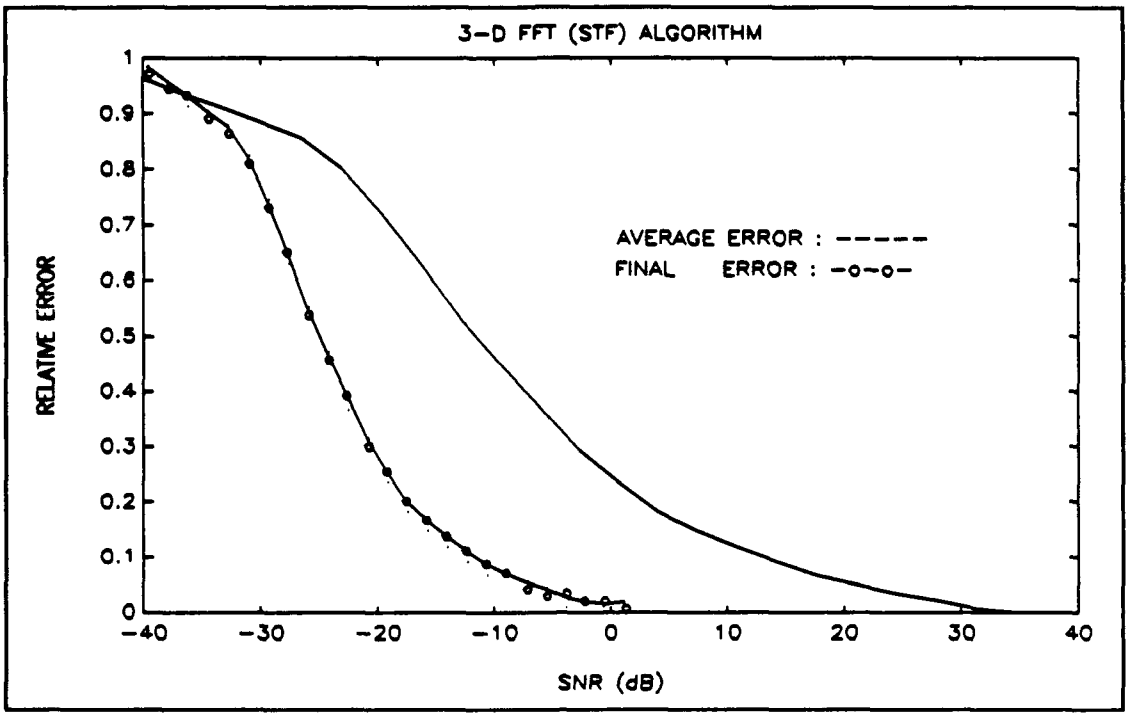


Figure 6.18 The Relative Error vs. SNR for 3-D FFT algorithm.

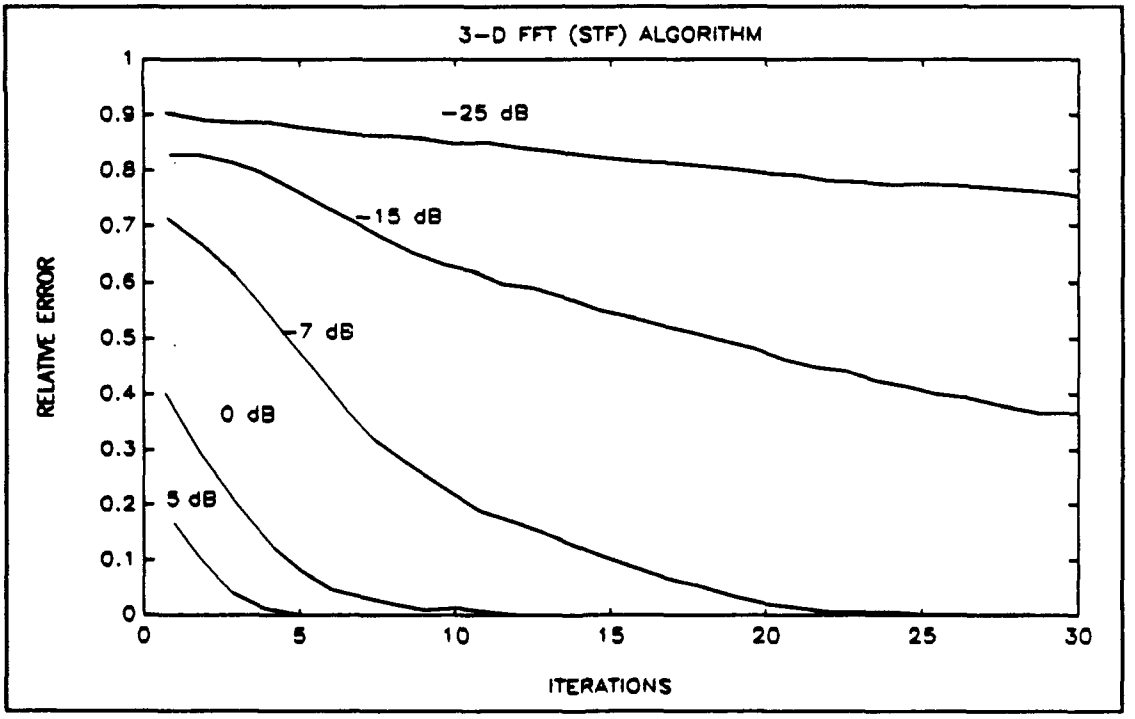


Figure 6.19 The Relative Error vs. Iterations for 3-D FFT algorithm.

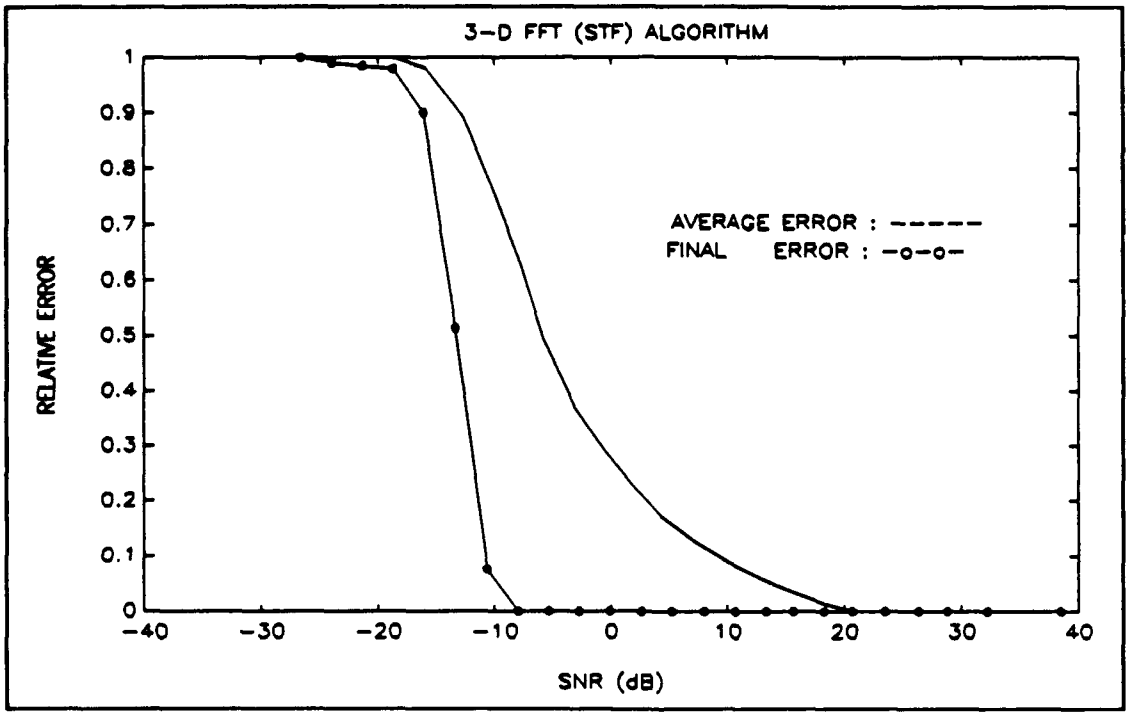


Figure 6.20 The Relative Error vs. SNR for 3-D FFT Algorithm (with background).

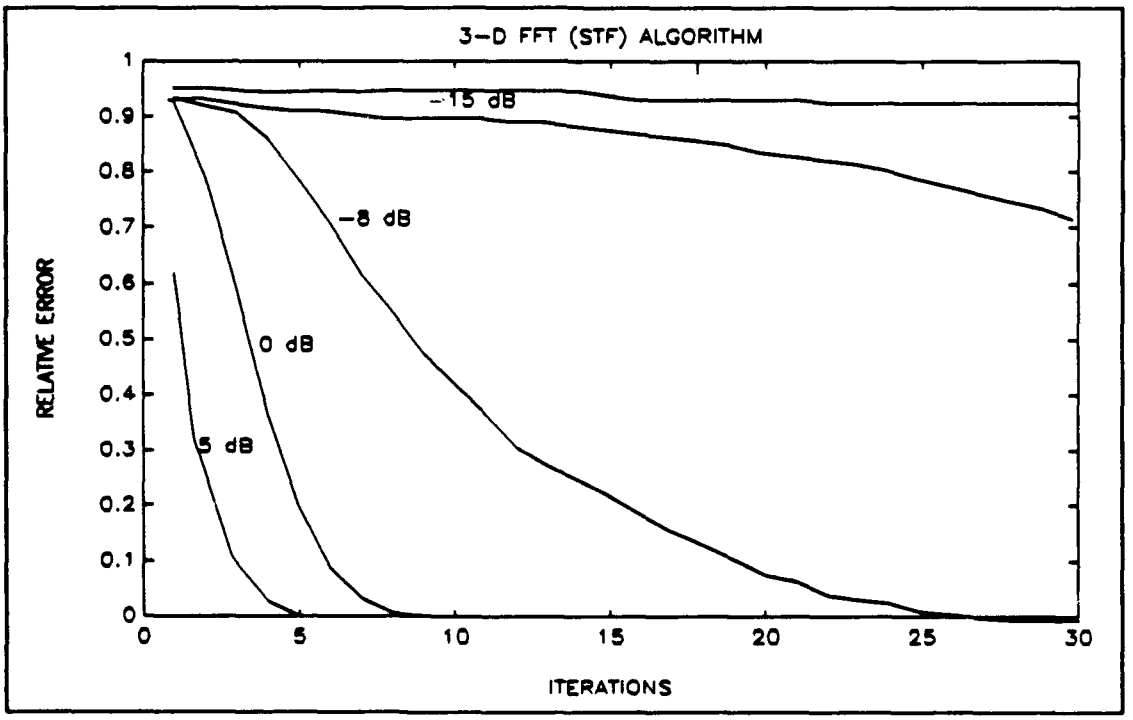


Figure 6.21 The relative Error vs Iterations for 3-D FFT algorithm (with background).

D. THE 1-DIMENSIONAL FFT ALGORITHM

This algorithm is the simplest, as mentioned before, but still gives fairly good velocity estimates. It takes much less computation time since it uses only one-dimensional Fourier transforms. Throughout the simulations, the estimates of this algorithm are also improved in two steps. First, the velocity components are found by averaging the results for 3 k_x values (k_x = 1,2,3). Second, the velocity components are computed by Spectral Estimation (MEM) methods as discussed in Chapter IV for k_x=1. The final velocity is estimated by comparing the MEM estimate with the average estimates. A sample test result is shown in Figure 6.22. For high SNR, even two frames are enough to yield a solution. But as the noise level increases more frames are needed. Overall this algorithm gives better results for the estimation of motion parameters than the other algorithms. The Relative Error vs. SNR and Iterations are plotted in Figures 6.23-6.26 for two choices of background.

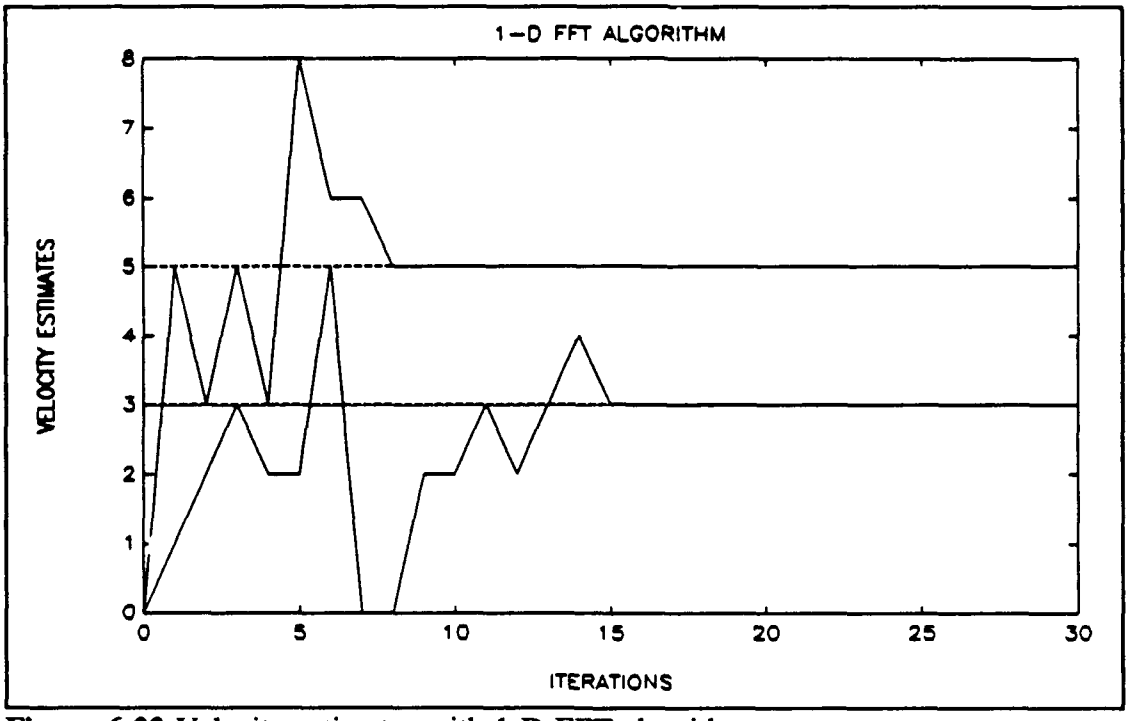


Figure 6.22 Velocity estimates with 1-D FFT algorithm.

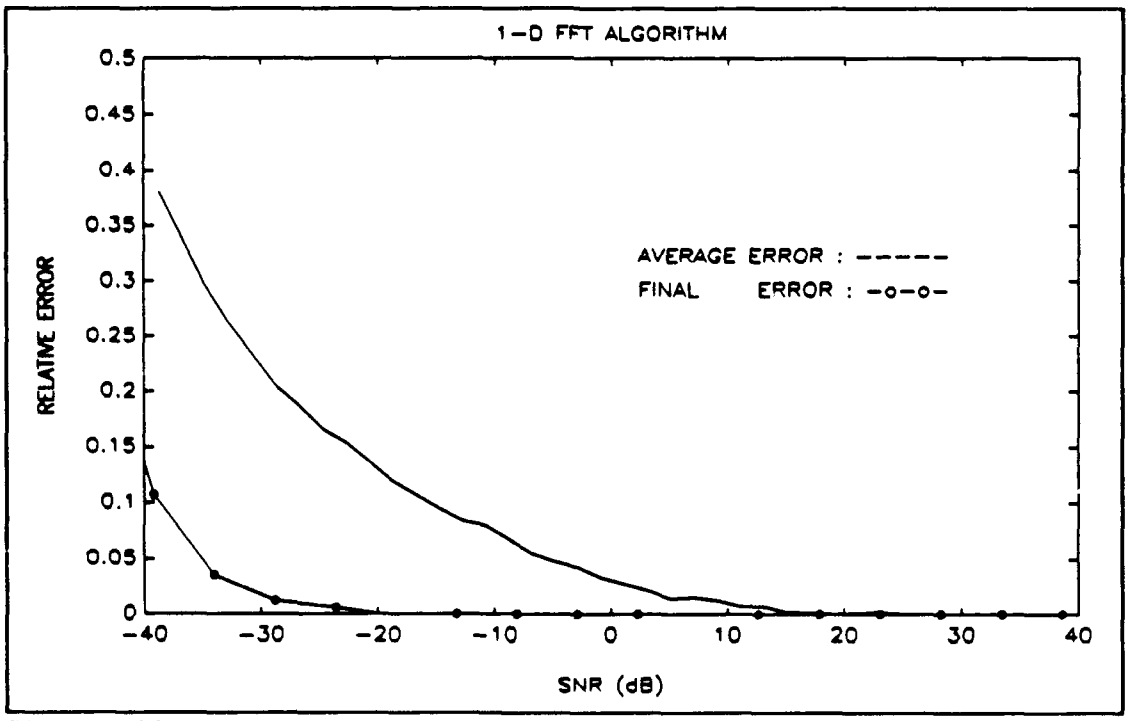


Figure 6.23 The Relative Error vs. SNR for 1-D FFT algorithm.

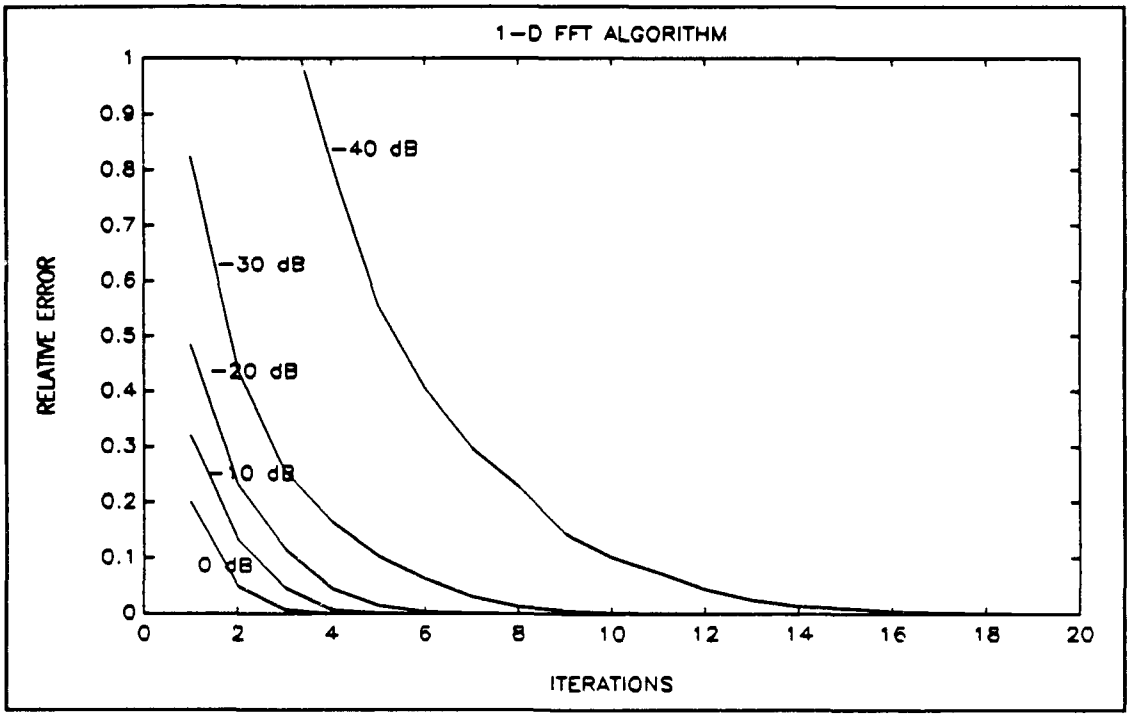


Figure 6.24 The Relative Error vs. Iterations for 1-D FFT algorithm.

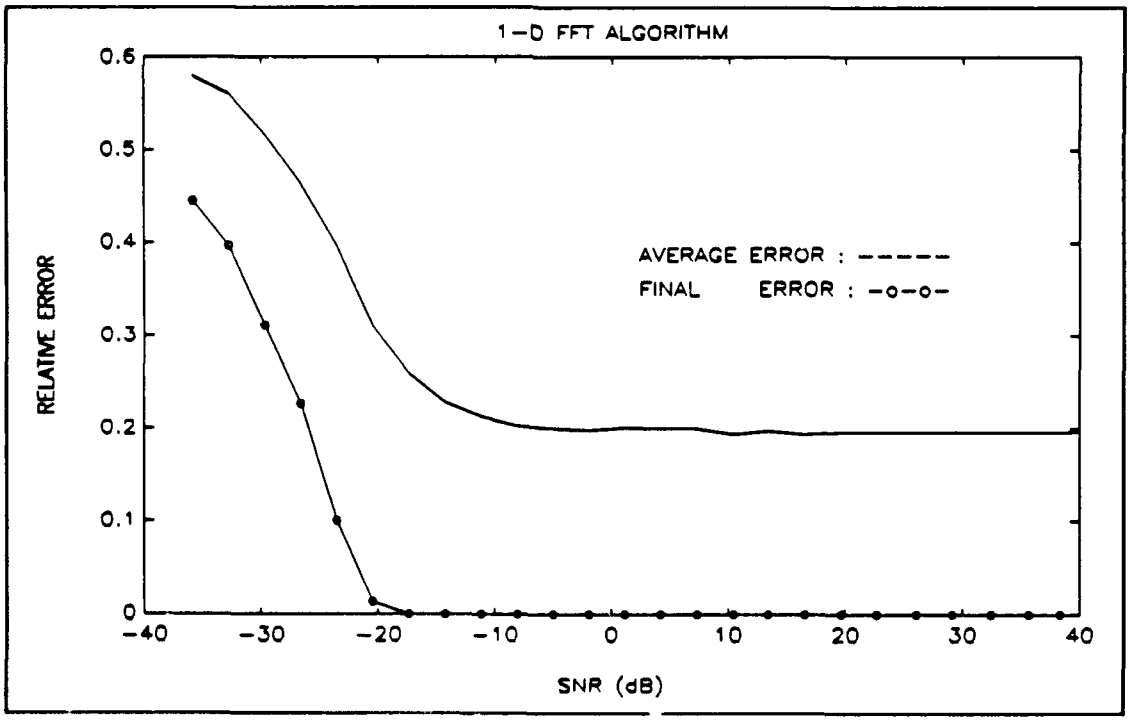


Figure 6.25 The Relative Error vs. SNR for 1-D FFT algorithm (with background).

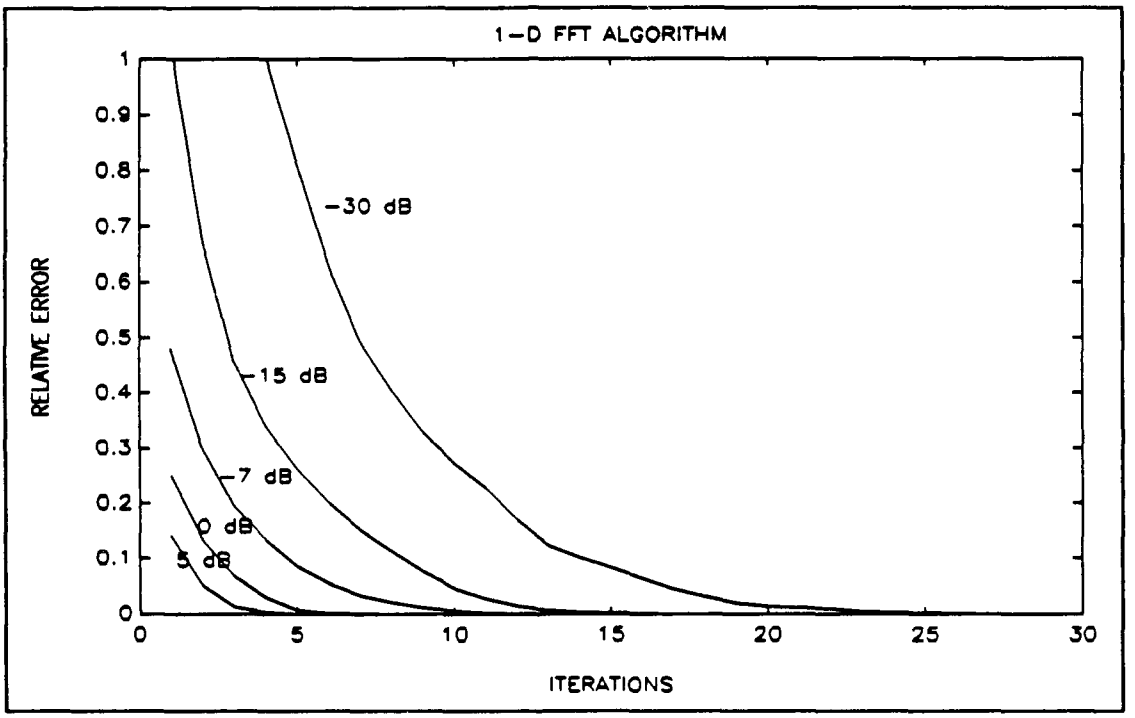


Figure 6.26 The Relative Error vs. Iterations for 1-D FFT algorithm (with background).

E. RESTORATION AND IMAGE ENHANCEMENT

The test image frames are restored by using the estimates of the EKF and by using the Velocity Tuned Filter (VTF) as outlined in Chapter V. The artificial images with decreasing SNR are used in the simulations. After each simulation, the SNR of the restored (or output) image frames are computed and compared with the SNR of the noisy (or input) image. The difference between these SNR values are plotted in Figures 6.30-6.31 to show the increase (or decrease) in the image quality. As seen from these plots the EKF does not improve the image quality for images with SNR's higher than 10 dB. This occurs since some of the higher spatial frequencies are truncated in the EKF algorithm. Similiarly, the results show an increase in image quality for images with SNR terms lower than -20 dB. However when the restored images are checked, it is seen that the increase in SNR is not a real increase in image quality, instead, it is an increase related to lowpass filtering of the image caused by truncating frequencies. The same effect is seen for VTF after -25 dB SNR's. Figures 6.26-6.29 show the original, noisy, and the restrored image frames of a sample run of these algorithms.

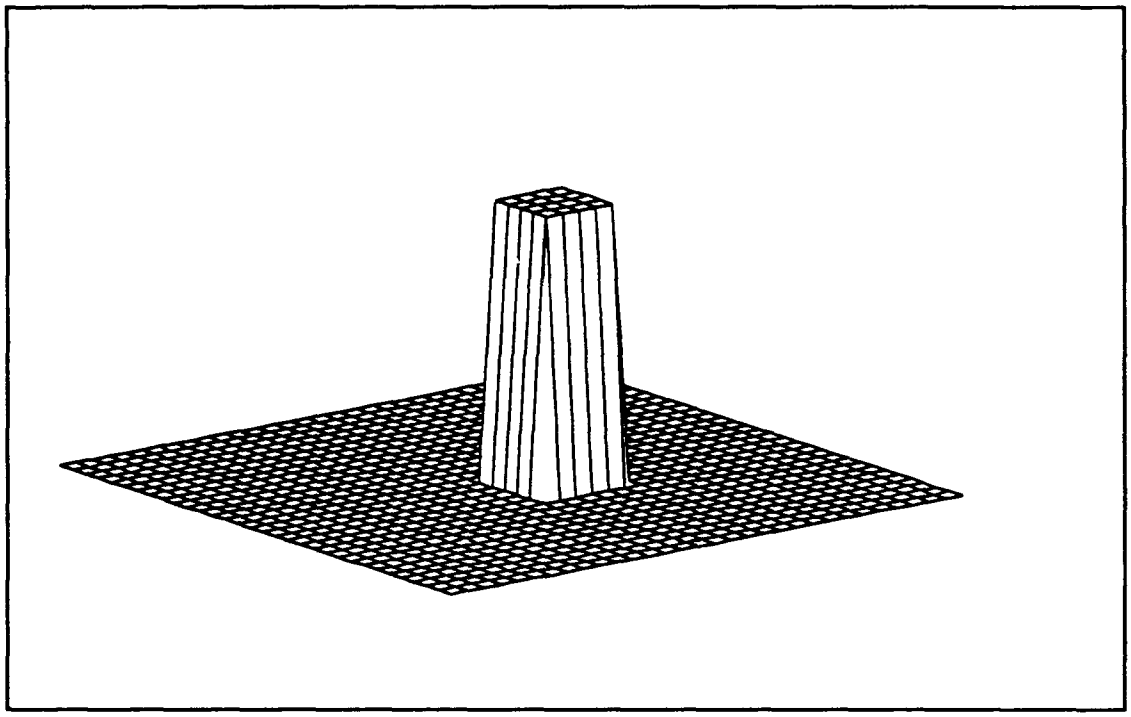


Figure 6.26 The original image frame without noise.

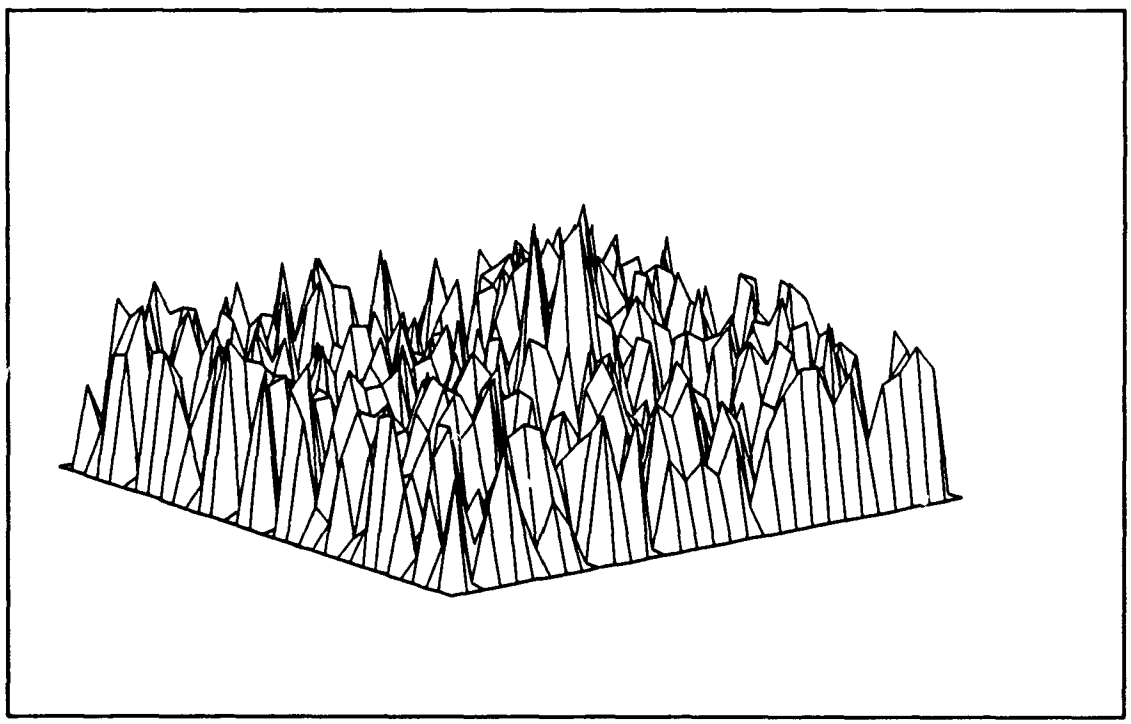


Figure 6.27 The original image with additive noise (SNR = -15 dB).

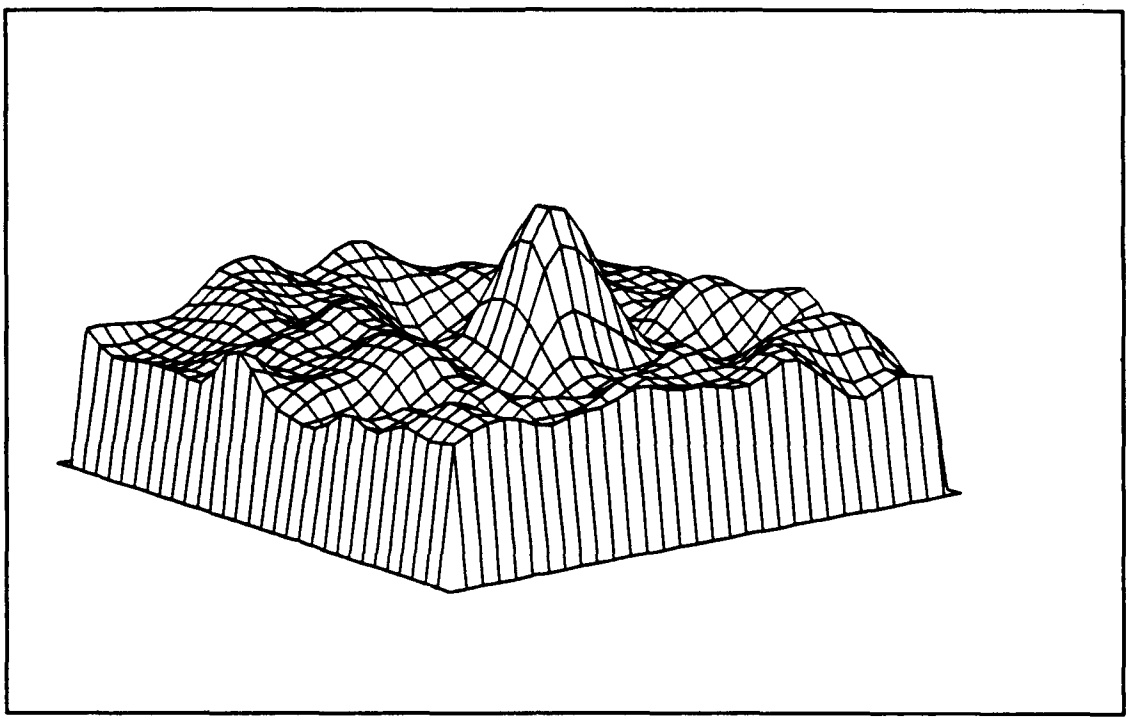


Figure 6.28 The restored image with EKF (SNR = -10 dB).

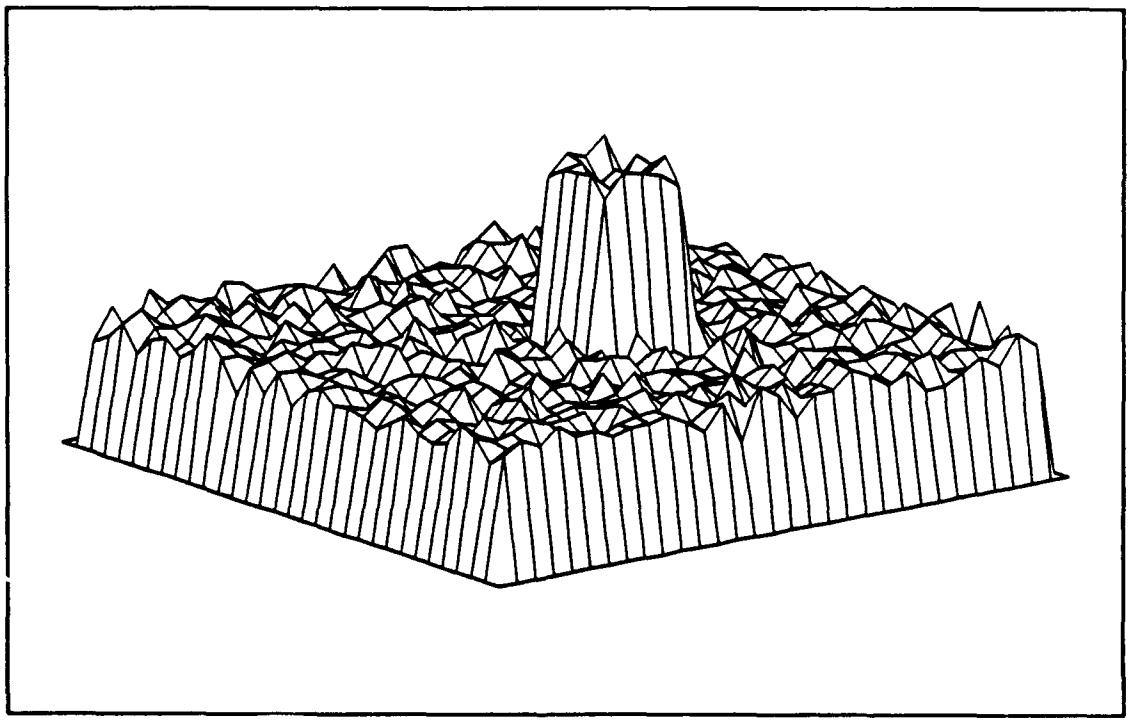


Figure 6.29 The restored image vith VTF (SNR = -7.5 dB).

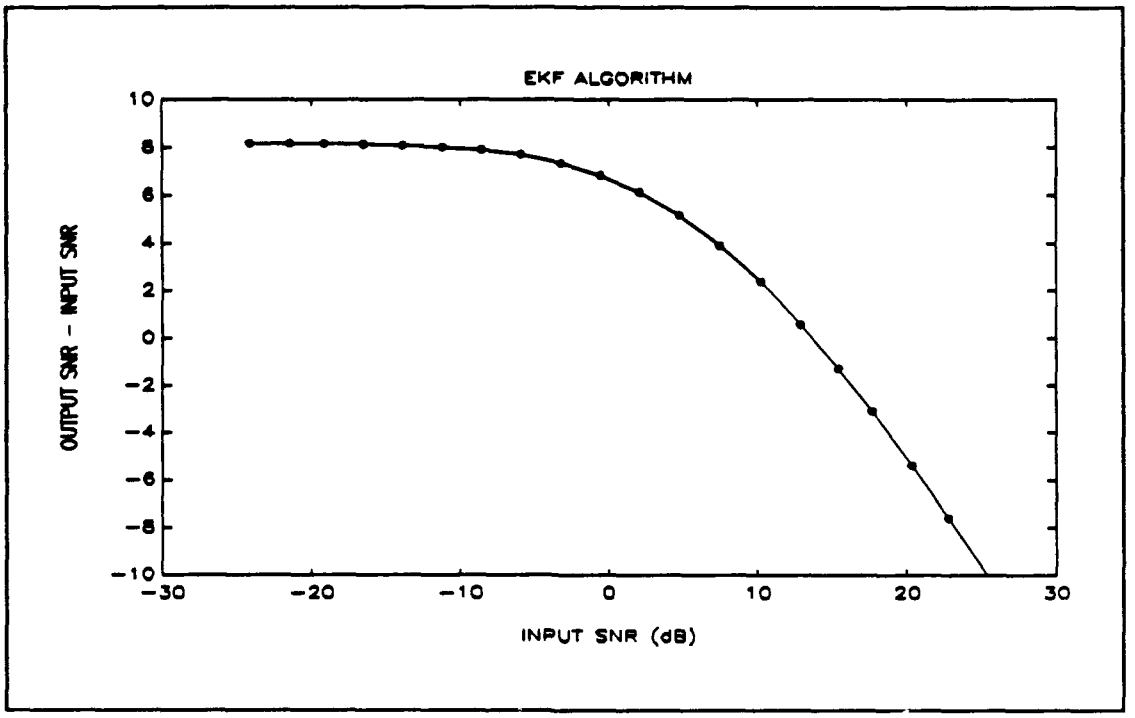


Figure 6.30 The increase in SNR with EKF.

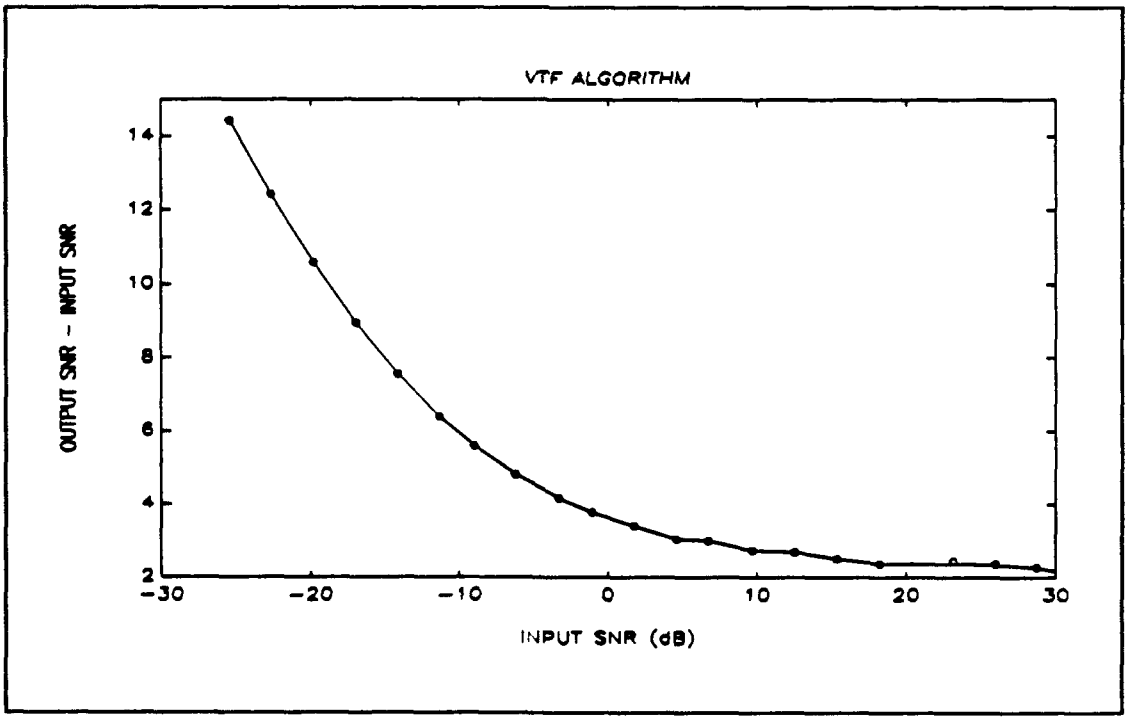


Figure 6.31 The increase in SNR with VTF.

F. CONCLUSIONS

As seen from the simulation results, the 1-D FFT algorithm gives the best result for the estimation of velocity components, although this algorithm does not preserve the original image. The EKF gives the second best result for the estimation of velocity components for low SNR images and also increases the image quality. The 3-D FFT (STF) algorithm does not perform as well as the others but promises an increase in the image quality with Velocity Tuned Filters (VTF). The algorithms that implement the Image Flow Constraint equation performed after smoothing. Additionally, these algorithms produce a velocity field besides only the estimation of a single two-dimensional velocity component.

Appendix A. Derivation of Image Flow Constraint Equation

Let E(x,y,t) be the irradiance at time t at the image point (x,y). Then if the image point is displaced a distance δx in the x-direction and δy in the y-direction in time δt, we expect that the irradiance will be the same at time $t+\delta t$ at the point $(x+\delta x, y+\delta y)$. So,

$$E(x,y,t) = E(x+\delta x, y+\delta y, t+\delta t). \tag{A.1}$$

The right hand side of Equation A.1 can be expanded about a point (x,y,t) using Taylor series and so obtain,

$$E(x,y,t) + \frac{\partial E}{\partial x}\delta x + \frac{\partial E}{\partial y}\delta y + \frac{\partial E}{\partial t}\delta t + \epsilon = E(x,y,t), \tag{A.2}$$

where ϵ contains second and higher order terms in $\delta x, \delta y$ and δt. After substracting E(x,y,t) from both sides and dividing through by δt, the result is,

$$\frac{\partial E}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial E}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial E}{\partial t} = \omega(\delta t), \tag{A.3}$$

where $\omega(\delta t)$ is a term of order δt that includes second and higher variations of x and y. In the limit as $\delta t \to 0$, the above equation becomes,

$$E_x u + E_y v + E_t = 0. \tag{A.4}$$

This equation is called the image flow constraint equation as mentioned before.

Appendix B. Fourier Transform of a Linearly Translating image

Let $f(x,y,t)$ represent the time varying image function that contains a uniformly translating object as given by Equation 3.1,

$$\begin{aligned} f(x,y,t) &= f(x-v_x t, y-v_y t) \\ &= f(x,y) * \delta(x-v_x t, y-v_y t) \end{aligned} \tag{B.1}$$

The 3-dimensional Fourier transform of $f(x,y,t)$ is given by,

$$F(w_x,w_y,w_t) = \int_{-\infty}^{\infty} \int f(x,y,t) e^{-j(xw_x + yw_y + tw_t)}\, dx\, dy\, dt. \tag{B.2}$$

Using the representation given in Equation B.1 gives,

$$F(w_x,w_y,w_t) = \int_{-\infty}^{\infty} \int \left[f(x,y) * \delta(x-v_x t, y-v_y t) \right] e^{-j(xw_x+yw_y+tw_t)}\, dx\, dy\, dt, \tag{B.3}$$

and

$$F(w_x,w_y,w_t) = \int_{-\infty}^{\infty} e^{-jtw_t} \left\{ \int \int_{-\infty}^{\infty} \left[f(x,y) * \delta(x-v_x t, y-v_y t) \right] e^{-j(xw_x+yw_y)}\, dx\, dy \right\} dt, \tag{B.4}$$

$$F(w_x,w_y,w_t) = \int_{-\infty}^{\infty} e^{-jtw_t} \left\{ F(w_x,w_y) \int \int_{-\infty}^{\infty} \delta(x-v_x t, y-v_y t)\; e^{-j(xw_x + yw_y)}\, dx\, dy \right\} dt. \tag{B.5}$$

Performing the integrals inside the brackets gives,

$$F(w_x,w_y,w_t) = \int_{-\infty}^{\infty} e^{-jtw_t} \left\{ F(w_x,w_y)\; e^{-j(v_x w_x + v_y w_y)t} \right\} dt, \tag{B.6}$$

and,

$$F(w_x,w_y,w_t) = F(w_x,w_y) \int_{-\infty}^{\infty} e^{-j(v_x w_x + v_y w_y + w_t)t} dt. \tag{B.7}$$

This yields the final result as,

$$F(w_x,w_y,w_t) = F(w_x,w_y)\, \delta(v_x w_x + v_y w_y + w_t). \tag{B.8}$$

LIST OF REFERENCES

1. J.B. Burl, *A Reduced Order Extended Kalman Filter for Moving Images,* unpublished manuscript, Naval Postgraduate School, Monterey, California, 1990.

2. D.E. Dudgeon, R.M. Mersereau, *Multidimensional Signal Processing*, pp. 61-67, Prentice-Hall, 1984.

3. B.G. Schunk,"The Image Flow Constraint Equation," *Computer Vision, Graphics, and Image Processing,* Vol.35, pp. 20-46, 1986.

4. B.G. Schunk, K.P. Horn, "Determining Optical Flow," *Artificial Intelligence*, Vol. 8, No. 2, 1981.

5. R.J. Schalkoff, *Digital Image Processing and Computer Vision,* Wiley, 1989.

6. D.E. Kirk, *Optimal Control Theory,* Prentice Hall, 1970.

7. B.G. Schunk, "Image Flow Segmentation and Estimation by Constraint Line Clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 10, pp. 1010-1024, 1989.

8. R.C. Gonzales, P. Wintz, *Digital Image Processing,* Addison Wesley, 1987.

9. S.A. Rajala, A.N. Riddle, W.E. Snyder, "Application of the One-Dimensional Fourier Transform for Tracking Moving Objects in Noisy Environments," *Computer Vision, Graphics, and Image Processing,* Vol. 21, pp. 280-293, 1983.

10. R.D. Strum, D.E. Kirk, *Discrete Systems and Digital Signal Processing,* Addison-Wesley, 1989.

11. S. Haykin, *Adaptive Filter Theory,* Prentice-Hall, 1986.

12. J.S. Lim, *Two-Dimensional Signal and Image Processing*, Prentice Hall, 1990.

13. H. Gafni, Y.Y. Zeevi, "A Model for Processing of Movement in the Visual System," *Bio Cybern,* Vol. 32, pp. 165-173, 1979.

14. L. Jacobson, H. Wechsler, "Derivation of Optical Flow Using a Spatiotemporal Frequency Approach," *Computer Vision, Graphics, and Image Processing,* Vol. 21, pp. 280-293, 1983.

15. J.G. Bliss, "Velocity Tuned Filters for Spatio-Temporal Interpolation," *IEEE Workshop On Motion (1986)*, pp. 61-66.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Chairman, Code EC Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	1
4.	Prof J. B. Burl, Code EC/Bl Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	2
5.	Prof R. Cristi, Code EC/Cx Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	1
6.	Deniz Harp Okulu K.ligi Kutuphanesi Tuzla, Istanbul/ TURKEY	2
7.	LTJG F. Ildiz TK. Navy Y. Goztepe Ressam Salih Erimez Sok. Bizim Apt. No:47/12 Kadikoy, Istanbul/TURKEY	3